centro per la produttività veneto

sede: camera di commercio i a. a. · corso fogazzaro, 37 · telefoni (0444) 26829 · 37105
36100 vicenza codice fiscale 80009830243 · partita iva 00656270246

Seminario su:

INTRODUZIONE ALLE TECNICHE DI AUTOMAZIONE

FLESSIBILE - ITP 83.2

"Sviluppo delle macchine di visione
per l'Automazione Flessibile"

D I S P E N S E

Relatore:

Dr. Ing. R. CASSINIS
c/o Politecnico di Milano
M I L A N O

ROBOT E SISTEMI SENSORIALI: L'INTERAZIONE PIU' DIFFICILE

Riccardo Cassinis

Sommario

Il problema della corretta connessione dei sistemi sensoriali ai robot è certamente il più difficile fra quelli che il progettista di tali macchine è chiamato a risolvere.

L'articolo esamina le difficoltà che si incontrano per una corretta ed efficace interazione fra sensori ed attuatori, e le prospettive offerte dalla tecnologia attuale, con particolare riguardo all'informatica.

1. INTRODUZIONE

Anche se la robotica industriale ha compiuto negli ultimi anni progressi giganteschi, molta strada resta ancora da percorrere per ottenere robot sufficientemente autonomi e facilmente programmabili.

Mentre i principali problemi relativi alla struttura meccanica e ai sistemi di controllo sono stati risolti, e non si prevedono per il prossimo futuro innovazioni sostanziali, molto resta ancora da fare per quanto riguarda il trattamento dell'informazione sensoriale, la cui corretta gestione rappresenta l'unico sistema per ottenere un significativo miglioramento delle prestazioni di queste macchine.

L'articolo esamina lo stato attuale dell'arte in questo campo e le prospettive per il futuro.

2. L'INTELLIGENZA NEI ROBOT.

La classificazione dei robot utilizzata dalla JIRA (Japan Industrial Robot Association) e, di riflesso, da molti costruttori di tali macchine, giapponesi e non, pone al livello più alto i cosiddetti "robot intelligenti".

Riccardo Cassinis - Progetto di Intelligenza Artificiale del Politecnico di Milano.

In realtà, la definizione è quanto mai inesatta, perchè include nella categoria macchine in grado di prendere sì decisioni in maniera autonoma, ma solo quando tutti i parametri relativi al come e al quando queste decisioni debbano essere prese siano forniti dal programmatore del sistema.

Facendo una analogia con i calcolatori, quanto sopra porterebbe a concludere che questi ultimi sono intelligenti per definizione, dal momento che includono tutti almeno una capacità decisionale.

Al contrario, l'ormai storico test di Turing per verificare l'esistenza o meno della capacità di pensiero di una macchina non è ancora mai stato superato pienamente, anche se qualche volta sono stati raggiunti risultati che lo soddisfacevano quasi del tutto [McCo79].

Per i robot il caso è ben diverso. I problemi dovuti alla interazione con un ambiente poco deterministico, la difficoltà di trasformare esattamente informazioni elettriche in corrispondenti azioni fisiche, e soprattutto l'impossibilità pratica di utilizzare convenientemente i segnali provenienti dagli organi di senso hanno fortemente rallentato la loro evoluzione verso macchine effettivamente dotate di capacità di pensiero.

3. I SENSORI NEI ROBOT ATTUALI.

I robot industriali dotati di sensori sono ormai moltissimi: alcuni di essi sono in grado di sfruttare molto bene le caratteristiche proprie dei sensori che li corredano. Occorre però notare che in tutti i casi i segnali provenienti dai sensori sono utilizzati o dal programma "di utente" , o dal sistema stesso per ragioni di sicurezza. Le azioni condizionate da parametri esterni che il robot è in grado di compiere sono quindi determinate da sezioni di programma che esso già contiene, siano esse "incorporate" nella macchina all'atto della sua fabbricazione, o introdotte successivamente dal suo utilizzatore.

Un esempio di quanto sopra si può trovare nel robot Supersigma, in dotazione al Laboratorio di Robotica del Politecnico di Milano, in cui i sensori presenti possono essere impiegati per entrambe le funzioni sopra citate [Cass81a].

Sempre con riferimento a questa macchina, sono state analizzate le prerogative di ciascuno dei sensori in essa utilizzati, e si è giunti alla conclusione che, nell'ambito della robotica industriale, è inutile e sbagliato classificare i sensori con gli schemi classici (tipo delle grandezze da misurare, trasduttori impiegati, ecc.), perchè il fatto stesso che questi sensori siano applicati a macchine così "strane" come i robot ne stravolge spesso completamente le possibilità di impiego. Una dettagliata analisi di questi concetti si può trovare in [Cass79]. In questo articolo si propone inoltre una classificazione basata sul tipo di uso che di ogni sensore si fa all'interno della macchina: questa classificazione permette di uniformare anche le modalità di gestione dei diversi tipi di sensori.

Si riporta qui per sommi capi la classificazione proposta. Le classi in cui vengono raggruppati **tutti** i sensori che possono essere collegati a un robot sono:

- **Classe A** - Sensori usati per il corretto funzionamento della macchina;
- **Classe B** - Sensori usati per garantire la sicurezza del robot e dell'ambiente circostante (operatori compresi);
- **Classe C** - Sensori usati per il corretto funzionamento del programma del robot.

Per la definizione di ciascuna delle classi citate (del resto piuttosto ovvia) si rimanda a [Cass81b].

Mentre è abbastanza semplice gestire efficientemente i sensori di classe A e B, sorgono enormi difficoltà nella gestione di quelli appartenenti alla terza classe. Essi possono essere infatti divisi in due sottocategorie, che chiameremo rispettivamente di **basso** e di **alto livello.**

La distinzione fra questi due ultimi tipi non è del tutto immediata: si può prendere come parametro discriminante la quantità di informazione che essi sono in grado di fornire.

Gli esempi che seguono chiariranno meglio il concetto.

3.1. Sensori di classe C di basso livello.

Consideriamo il sistema di visione di cui è attualmente dotato il Supersigma, che è descritto in dettaglio in [Cass81c]. Questo apparecchio è stato progettato per mettere in grado i due robot del laboratorio di afferrare "con cognizione di causa" oggetti portati sotto le loro mani da un nastro trasportatore. In altre parole, il sistema è in grado di riconoscere il tipo e l'orientamento di oggetti che gli siano stati preventivamente "mostrati". Ora, per quanto complesso possa essere il sistema, la quantità di informazione che esso fornisce al sistema di controllo dei robot è contenuta in un numero assai limitato di bit. Inoltre, queste informazioni devono essere gestite dal programmatore, o mediante istruzioni contenute nel programma del robot, o (ma non è del tutto semplice) mediante routines predefinite dal fabbricante o dallo stesso programmatore.

Siamo quindi ovviamente in presenza di un sensore di classe C (serve infatti per ottenere un corretto funzionamento del programma del robot), ma di **basso livello,** perchè le informazioni fornite sono poche, e devono essere gestite dal programmatore. Inoltre, queste informazioni sono gestite esattamente nello stesso modo in cui si gestiscono quelle provenienti da altri sensori estremamente più banali (di prossimità, di forza, ecc.). Giova ripetere ancora una volta che tutto ciò è assolutamente indipendente dalla complessità intrinseca del sensore.

3.2. Sensori di classe C di alto livello.

Consideriamo ora un ipotetico sistema che metta in grado il robot di scegliere automaticamente le traiettorie da seguire durante gli spostamenti da un punto a un altro, basandosi sulla osservazione visiva del campo di lavoro del

robot stesso.

E' evidente che in questo caso occorre una gestione completamente diversa da quella appena descritta: l'unica soluzione ragionevole appare essere quella in cui il programma che descrive i movimenti del robot viene generato in tempo reale sulla base delle informazioni visive ricevute istante per istante. L'esperienza mostra che è possibile utilizzare altri metodi solo in casi molto particolari, del tutto al di fuori della generalità e della flessibilità che si richiedono a un robot degno di questo nome.

4. GESTIONE DI SENSORI DI CLASSE C DI ALTO LIVELLO.

L'unico tentativo organico di gestire efficientemente questi sensori è quello proposto da Albus e Barbera [Alb79 e Alb81]. La proposta consiste nel suddividere il sistema di controllo in diversi livelli gerarchici, a ognuno dei quali è demandata una particolare funzione dell'algoritmo globale di controllo, e nel suddividere l'informazione sensoriale in analoghi livelli, in maniera tale che ad ogni livello gerarchico di controllo siano presentate solo le informazioni sensoriali che gli competono.

Concettualmente, l'idea non è molto diversa da quella presentata in [Cass79], con la differenza che, mentre nel Supersigma le informazioni vengono "smistate" alla fonte, in quello proposto da Albus e Barbera esse viaggiano su un bus comune, e ad ogni livello vengono "catturate" solo quelle che gli competono.

Inoltre, nell'ultima implementazione di questo sistema, è stato aggiunto ad ogni livello un blocco di rappresentazione dello stato del mondo, che integra le informazioni sensoriali, permettendone una gestione molto più efficiente.

Questo sistema è indubbiamente dotato di una elevatissima originalità, in quanto rappresenta la prima applicazione pratica delle idee esposte nel par. 3.2. Le sue prestazioni, a quanto si può desumere da un documentario che è stato proiettato all'I.S.I.R. di Tokyo nel 1981, sono estremamente interessanti, perchè la macchina controllata da questo sistema è in grado di far fronte a situazioni assolutamente imprevedibili (per esempio, pezzi che si spostano mentre si cerca di afferrarli).

Lo svantaggio più grande è invece nella estrema difficoltà di programmare in maniera semplice una macchina di questo genere, dal momento che, allo stato attuale dell'arte, non è ancora possibile limitarsi alla descrizione "generica" delle operazioni da compiere, ma occorre anche fornire dettagliate istruzioni ai livelli intermedi del sistema di controllo.

5. CONCLUSIONI.

La strada da percorrere per ottenere un significativo miglioramento delle prestazioni dei robot, che si può avere solo perfezionando la tecnica della gestione delle informazioni sensoriali, è certamente ancora molto lunga, e irta di problematiche pressochè inesplorate.

I prevedibili progressi dell'Intelligenza Artificiale lasciano però sperare che, in un futuro probabilmente non troppo lontano, sistemi "intelligenti" potranno essere sostituiti a quelli attualmente impiegati, consentendo così quel passo in avanti che permetterà la trasformazione degli attuali robot, che non sono altro che macchine a controllo numerico del tipo più avanzato, in qualcosa di completamente nuovo e rivoluzionario.

BIBLIOGRAFIA

[McCo79] P. McCorduck
Machines Who Think - Pag. 225 e seg.
Freeman and Company
San Francisco, 1979.

[Cass81a] R. Cassinis
An Example of a Distributed Intelligence Discrete Process Controller
Digital Systems for Industrial Automation
Volume 1 N. 1 - Gennaio 1981 - Pag. 45-77.

[Cass79] R. Cassinis
Sensing System in Supersigma Robot
Proc. 9th ISIR - Washington, 1979.

[Cass81b] R. Cassinis
La sensorialità nei robot: problemi e prospettive
Automazione e Strumentazione
Volume XXIX - Aprile 1981 - Pag. 299-301.

[Cass81c] R. Cassinis, L. Schnickel
Il sistema di visione del robot Supersigma
Atti Congresso Aica 1981 - Pavia, 1981.

[Alb79] J.S. Albus, A.J. Barbera, M.L. Fitzgerald
Hierarchical Control of Robots Using Microcomputers
Proc. 9th ISIR - Washington, 1979.

[Alb81] J.S. Albus, A.J. Barbera
Hierarchical Control of Robots
Proc. 11th ISIR - Tokyo, 1981.

Sensing System in an Experimental Robot

Riccardo Cassinis, Politecnico di Milano, Milano, Italy

Abstract

The array of microcomputers that controls an experimental robot has made it possible to implement a sophisticated sensing system that is used for a wide variety of purposes.

The aim of the paper is to show how comparatively simple and inexpensive sensors, in cooperation with the great computing power of the microcomputers array and with a suitable programming language, may be used for jobs that are well beyond their normal operating capabilities. Included is an explanation of how complex sensors (such as vision systems) may be connected to the system in a simple and straightforward way.

Keywords: *Multicomputer Arrays, Sensing Systems, Sensors, Industrial Robots.*

Much is being done to improve the performance of industrial robots with the addition of sensors to their control systems.[1,2] This paper describes the work that has been done in this direction at the Department of Electronics of the Politecnico of Milano.

The robot shown in *Figure 1* is an experimental robot mainly designed for assembly operations.*

The sensing system that has been added to the robot greatly improves its overall performance, since it allows conditional branches in the programs, i.e., the capability of taking autonomous "decisions" about the tasks the robot is performing.

The paper shows how sensors may be used in an industrial robot without increasing the complexity of the machine, provided that a distributed intelligence system is available. This is especially important when the control system has to be modular. In this case, the addition of new sensors does not require any change in the hardware, but only that some new modules be installed.

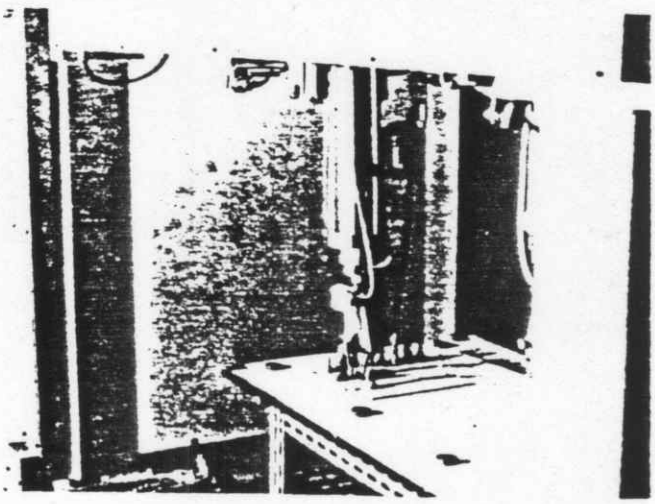


Figure 1
The Mechanical Part of the Robot

* The mechanical part of the machine is the Olivetti SIGMA feasibility model[3] that was donated to our research group in 1975. The control system on the other hand, was completely designed and implemented by researchers in the Department of Electronics of Politecnico di Milano.

Since data communication is more or less the same for all modules, base software also requires only minor changes.

Sensors in the Experimental Robot

The first step in approaching sensor problems in the experimental robot was an accurate analysis of their functions and characteristics. This analysis has led to the conclusion that, besides the nature of sensors, they may be divided into the following three categories:

1. Sensors that are used for proper functioning of the robot.
2. Sensors that are used for safety purposes, both of the machine and of the surrounding environment.
3. Sensors that are used for proper functioning of the robot program.

The same sensor may belong to more than one class at the same time. For instance, some microswitches are used in the robot to determine the initial position of the arms (class A), and to avoid crashes against the end-of-rail stops (class B). Force sensors belong to class B (the arms stop if the hands hit unexpected objects), and to class C (e.g., when they are used to measure forces during assembly operations).

It is also important to notice that class B sensors are always used as binary devices. This means that signals coming from analog sensors must be compared with a preset threshold, and that the result of this comparison is the only significant information for this class of sensors. Moreover, only one of their possible states is significant, since the other one indicates normal operating conditions. For this reason, signals from these kinds of sensors may be regarded as flags.

The classification just made is extremely important when the control system is based on hierarchical principles, as it happens to be in the experimental robot. This is because sensor information follows different paths.

Connecting Sensors to the Robot

The robot control system[4] is based on an array of microcomputers (*Figure 2*) that receive commands from a Central Control Unit (CCU), process them and feed transducers that drive actuators, or that receive signals from sensors.

The structure of the array is quite general, and does not depend on the nature and number of actuators and sensors employed. The only differences that may be found between microcomputers that drive different devices are in firmware and in transducer interfaces. The whole communication path between the CCU and the microcomputers will be hereafter referred to as MICROBUS.

Microcomputers receive commands from the CCU through a common bus (commands bus in *Figure 2*). Each command describes an elementary task to be performed by a microcomputer, as, for instance, "move axis x of left arm 500 steps in forward direction". Each command is 5 eight-bit words long, and contains the number of the microcomputer it is directed to, and concise description (in terms of op-codes and data) of the task to be performed.

Status lines are used to provide the CCU with information about the status of each microcomputer, so that commands may be issued at the proper time, according to the program running in the CCU.

Signals coming from class A and B sensors are directly fed into the microcomputers. This is because the program running in the CCU is typically a user program, and may therefore contain errors, while microcomputers firmware is never changed, and may be considered error-free. If the protection system were implemented at the CCU level, it would be more complex, both from the hardware and the software point-of-view.

The signal path just described implements a sort of nonlinear feedback, that will be hereafter called "short-loop feedback".

On the other hand, signals from class C sensors reach the CCU by means of a common data bus (answers bus in *Figure 2*). In order to avoid conflicts between microcomputers that send their data to the CCU, a protocol based on a "speak when you are requested to" principle was implemented.

In other words, microcomputers that drive class C sensors must receive a special command from the CCU before sending data to it. This also simplifies handling of sensors that are multiplexed into a single A/D converter, as it happens for force sensors in the robot. Since data from class C sensors

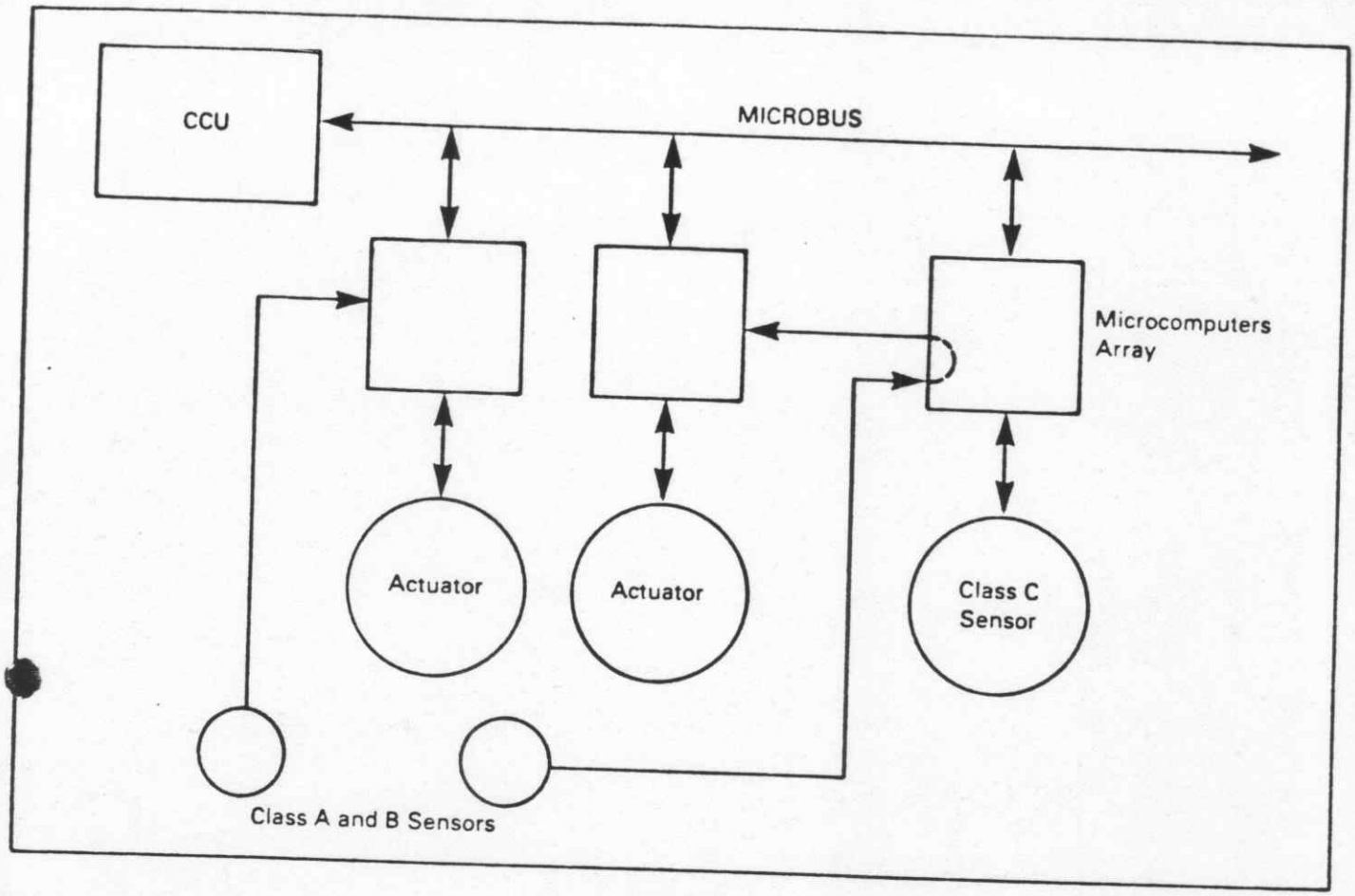


Figure 2
Block Diagram of the Robot Control System

usually modify the execution of the CCU program, the signal path just described is called "long-loop feedback".

Short-Loop Feedback Network

The purpose of this network is to avoid situations which could be dangerous for the machine or for the surrounding environment, and to obtain proper performance from the robot.

No positional feedback exists in the robot, since all arm movements are obtained by open-loop stepping motors. For this reason, limit switches are used to determine an initial reference for the arms position.

Once the origin has been found (this is obtained by moving each motor in the "backwards" direction, until the limit switch is activated, then in the opposite direction until it is released). Any other activation of the limit switch may only mean that an erroneous command was issued by the CCU, or that the actual position of the arms is different from the computed one, possibly because a loss of synchronism in the motors has occurred.

In both cases, the motor that drives the arm in the direction along which the end of rail was reached must be stopped, and this is immediately done by interrupting the microcomputer that drives the motor. The microcomputer then executes a special stopping routine, whose characteristics depend upon the motor being used. When the motor is still, the CCU must be informed of the abnormality that has occurred. This is done by setting appropriate bits of the status lines that reach the CCU.

Limit switches are used as class A sensors only when origination of the arms is being performed. In such case, their activation is expected, and no abnormal status signal is issued.

The other sensors that are presently connected to the short-loop feedback network in the experiment are force sensors mounted on the wrists. In order to use them as class B sensors, their output must be continuously read, converted to a digital quantity, and compared with two preset thresholds. If the

value read from one sensor overcomes one of the two thresholds, the motor driving the arm in that direction must be stopped, since this means that the hand has probably hit an unexpected object.

The microcomputer that drives the force sensors then interrupts the concerned motor microcomputer. The stopping routine is similar to the one used in case of end of rail emergencies, but it stops the motor in a shorter time, since the force acting on the hand may be used as an additional brake. As it happens for end of rail emergencies, the CCU is informed of the abnormal stop by means of the status lines. A block diagram of the force sensors system is shown in *Figure 3*.

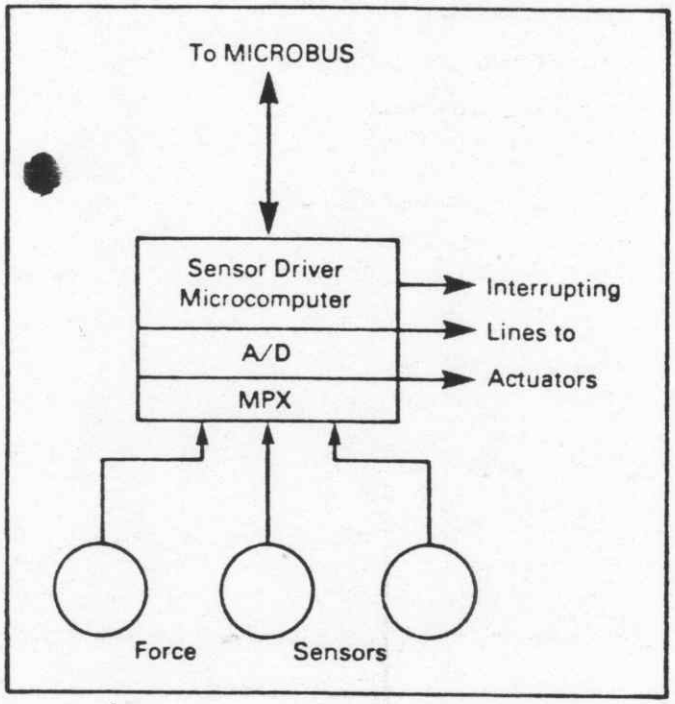


Figure 3
Block Diagram of Force Sensors System

Actions performed by the CCU upon reception of an abnormal status, depend on the program running on it. In Multipurpose Assembly Language[5] (MAL), i.e., the programming language presently being used to program the experimental robot, any abnormality vectors the program to a predefined location, where a programmer written emergency routine must be stored. The vectors are programmable, so different actions may be undertaken in accordance with the various emergencies encountered.

When operations are expected that require a force to be issued from the hands of the robot, the preset thresholds may be changed by the CCU, so that no interruptions on the axis performing the task may occur. Changing force thesholds is simply done by sending a command to the microcomputer that drives the force sensors.

While the end of rail protection has been thoroughly tested, and has shown that no damage may occur to the machine even if wrong commands are deliberately issued from the CCU, the force protection system is now being tested.

The first results are quite good from an electronic point-of-view. Testing the six force sensors, converting their analog signals to digital and comparing the signals with the two thresholds requires about one millisecond. This means that in the worst case, a crash of the hand will be detected one millisecond after it has occurred. Since the maximum travelling speed of the arms is one metre per second, the hand will move one millimetre (in the worst case) before the crash is detected. This is not a problem, since the wrist is flexible, and it may accomplish a displacement of much more than one millimeter between the hand and the arm. The problem is that the motor cannot be stopped instantaneously, and this increases the actual travel of the arm before it comes to a complete stop.

This problem may be solved in two ways, (1) use electromagnetic brakes on the motors, and (2) use proximity sensors instead of force sensors. The first solution, however, does not seem applicable, because it would cause stresses in the transmission system that could impair its precision. The second one seems more interesting, although some minor problems are still to be solved.

The behaviour of short-loop feedback may be compared to the reflex arc phenomenon that may be found in many animals, man included. If, for instance, a hand is placed on a hot object, it is usually retracted before pain is felt. This happens because stimuli coming from sensorial organs are processed at ganglions level, and the result of this processing causes hand retraction. The brain is not involved in this process, and knows that something has happened only after "emergency recovery" actions have been undertaken.

Long-Loop Feedback Network

During the execution of a robot program, it is often necessary to measure parameters of the world surrounding the robot that may vary in a non-deterministic way. The kinds of sensors that may be

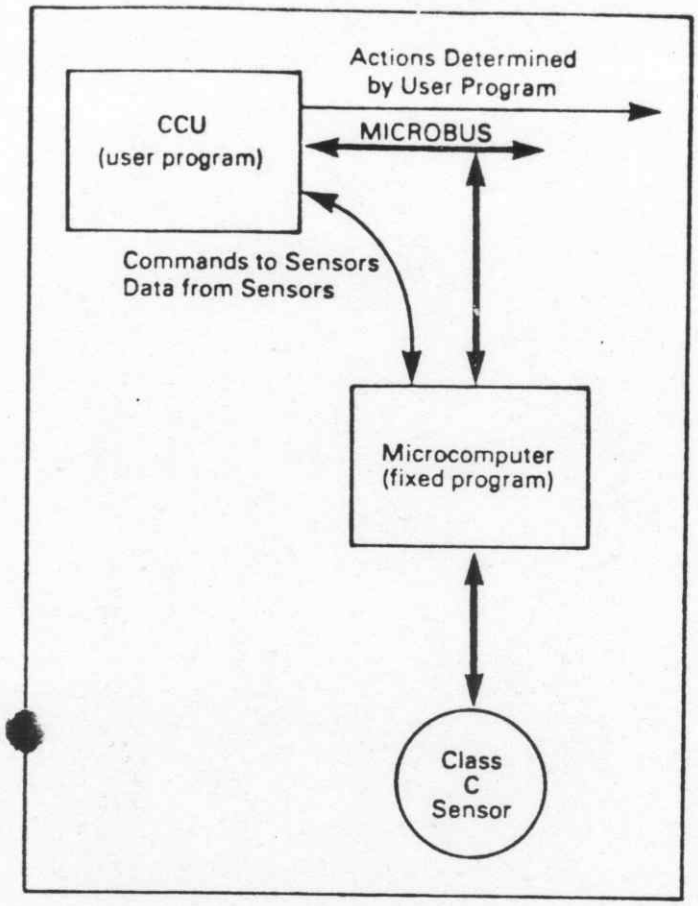


Figure 4
General Structure of Class C Sensors Connection to the Control System

used for such purposes vary from extremely simple sensors, as force sensors, to extremely complex devices, as vision systems. In order to avoid difficulties arising when new sensors are added, the structure of the system that drives the robot does not take into account the nature of sensors, just as it happens with actuators.

The general structure of a class C sensor connection to the system is shown in *Figure 4*. The transducer employed depends on the kind of sensor being used, while the microcomputer has the task of receiving commands from the command bus and of sending appropriate answers on the answer bus.

With this sort of connection, very complex sensors are seen from the system just as the simplest ones. For instance, let us consider the vision system that is now being implemented.

The main purpose of this vision system is to 'see' objects being carried to the robot by a conveyor belt. The position of these objects is unknown, and the system must be given information about their position in order to correctly pick up the objects.

Although processing of an image is always quite a complex task, data that must be transmitted to the CCU refer only to the position of each object, i.e., its centroid coordinates, the angular displacement of the object with respect to a known direction, and an 'upside-down' indicator. This of course applies only to flat objects, that may be completely identified by examining their shadow.

In case of more complex shapes, the vision system becomes much more complicated, but parameters that define the kind of position of any object are only seven, and only these parameters need to be transmitted to the CCU.[6] A single microcomputer is then required to connect the vision system to MICROBUS, and the communication protocol is the same as for force sensors.

On the other hand, if sensors are required to strictly interact with the system (e.g., in a visually-aided assembly machine), it seems more convenient to interface these sensors directly to the CCU, so that MICROBUS will not be overloaded by the data flow between the CCU and the vision system. It must be kept in mind, however, that MICROBUS capacity is in excess of 10,000 commands per second, i.e., over 400 KBaud. Another interesting example is how simple sensors may be used in robots to provide information beyond their normal capabilities.

A lamp-photodiode couple was mounted in the fingers of the robot (*Figure 5*). The original purpose

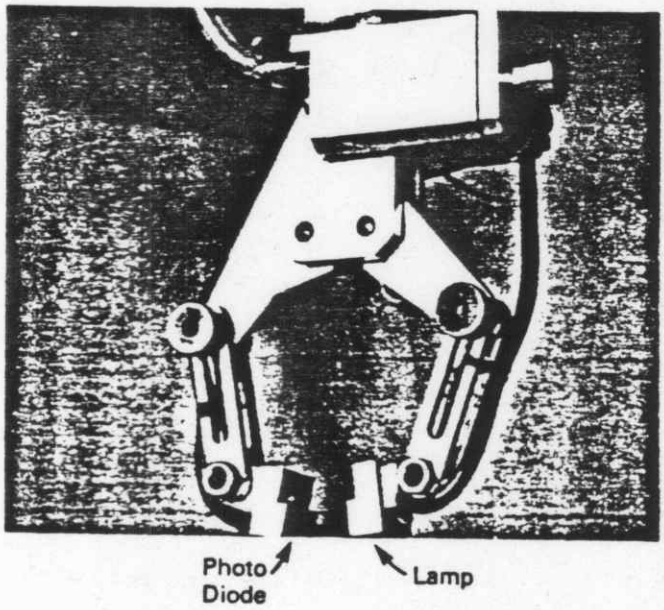


Figure 5
The Robot Hand and Object Detector

was to verify if an object was present in the hand at some points of assembly operations. It was noted however, that the same sensor could be used for more sophisticated operations, such as the lookup of pieces, or even the measurement of their dimensions. This is possible because coordinates of the hand are always known, and by combining a few MAL instructions the task may be easily solved.

As an experiment, a program was written that looked for chess pieces randomly placed along some lines on the working board of the robot, measured their height, and placed them accordingly on other lines.

The most obvious application of this sensor is to eliminate some parts feeders, or to substitute them with simpler ones, since many objects no longer require an exact initial position.

Implementation

The first implementation of the system was based on an array of National Semiconductors SC/MP microprocessors. The microprocessors comparatively low speed caused some problems with force sensors, since, as previously stated, sensors must be continuously tested in order to determine if forces are within the thresholds.

The system has now been rebuilt using INTEL 8748 single-chip microcomputer to reduce the component count, thus increasing the reliability of the machine. The introduction of 8748 also allows a more sophisticated performance of the array, since commands and answers busses may be combined into a single 8-bit bus, thus allowing any microcomputer to become master of the system. This solution was not convenient in the old implementation. Furthermore, 8748 is about eight times faster than SC/MP, which solved some problems related to the low computing speed of SC/MP.

Experimental Results

Some of the advantages and drawbacks of the system were previously described. The overall performance of the system is very good in comparison to similar robots.

Handling of sensors is quite easy in MAL, since a number of standard instructions are provided for standard sensors, and a small set of general purpose instructions is available for new sensors that are being added to the system from time to time. If these sensors prove to be useful, dedicated instructions are defined and inserted in MAL.

Some practical problems arise with force sensors that are differential magneto-resistors, whose displacement with respect to permanent magnets fixed to the flexible wrist is proportional to the force applied to the hand.

However, we believe that systems based on strain gages or on conductive rubber would allow a greater precision and would reduce the mechanical hystheresis of the actual system, which is particularly noticeable after strong shocks. A self-calibration procedure has however been included in the microcomputer program, in order to allow compensation for these mechanical drawbacks.

Apparent forces due to accelerations and decelerations may be compensated by small accelerometers mounted on the wrists. Signals from these accelerometers would be used to correct force readings by simple analog operations. In the experimental robot, the mass of the hands and the usual payloads are quite small, and these corrections do not seem to be necessary. Once the vision system is ready, it will probably cause no problems for the connection to MICROBUS, since it will work as force sensors do.

There are no problems with respect to MICROBUS capacity, since it has been found that it is currently being used only at about .15% of its capacity in the most extreme case. This means that the system may be greatly expanded to include other robots or special-purpose devices that may be driven by the same CCU, thus greatly simplifying cooperation and coordination problems.

Concluding Remarks

The sensing system that has been presented is only a part of the research on robots that is being carried on at the Milan Polytechnic Artificial Intelligence Project. It is obvious that the sensing system is tailored according to the needs of our robot, and that it may work only if an array of microcomputers as the one described is present. It seems however, that the theory discussed in this paper could also be useful for robots that are driven by different systems.

Acknowledgement

The author wishes to acknowledge all those researchers and students who have contributed their work and ideas to the project and to the realization of the robot and of this paper.

This research was partially sponsored by the National Research Council.

References

1. A. J. Barbera, J. S. Albus and M. L. Fitzgerald, "Hierarchical Control of Robots Using Microcomputers", *Proceedings of the 9th International Symposium on Industrial Robots*, Washington D. C., 1979, pp. 405-422.

2. A. J. Barbera, J. S. Albus and M. L. Fitzgerald, "Hierarchical Control for Sensory Interactive Robots", *Proceedings of the 11th International Symposium on Industrial Robots*, Tokyo, 1981, pp. 497-505.

3. A. D'Auria and M. Salmon, "SIGMA: An Integrated General Purpose System for Automatic Manipulation", *Proceedings of the 5th International Symposium on Industrial Robots*, Chicago, 1975, pp. 185-202.

4. R. Cassinis and L. Mezzalira, "A Multimicroprocessor System for the Control of an Industrial Robot", *Proceedings of the 7th International Symposium on Industrial Robots*, Tokyo, 1977, pp. 235-242.

5. G. Gini, M. Gini, R. Gini and D. Giuse, "Introducing Software Systems in Industrial Robots", *Proceedings of the 9th International Symposium on Industrial Robots*, Washington D. C., 1979, pp. 309-321.

6. K. Tani, M. Abe, M. Tani and T. Ohno, "High Precision Manipulator with Visual Sense", *Proceedings of the 7th International Symposium on Industrial Robots*, Tokyo, 1977, pp. 561-568.

Author Biography

Riccardo Cassinis was born in Milano, Italy, and graduated in Electronic Engineering in 1977. Since then, he has worked at the Department of Electronics of Politecnico di Milano as a member of Milan Polytechnic Artificial Intelligence Project. His present position is Research Associate.

In 1975, Mr. Cassinis started the design of the control system for the experimental Supersigma robot, which was the first application of a multimicrocomputer system to control an industrial robot. For this work, he was awarded first place in the 1981 Eurocitema contest, which awarded the best application of mini and microcomputers to robotics done by European citizens under the age of 30.

His professional interests now include robot control systems, programming languages and robot sensing systems. He is also interested in other computer applications, such as home automation and special-purpose mechanical machines.

1961

1981

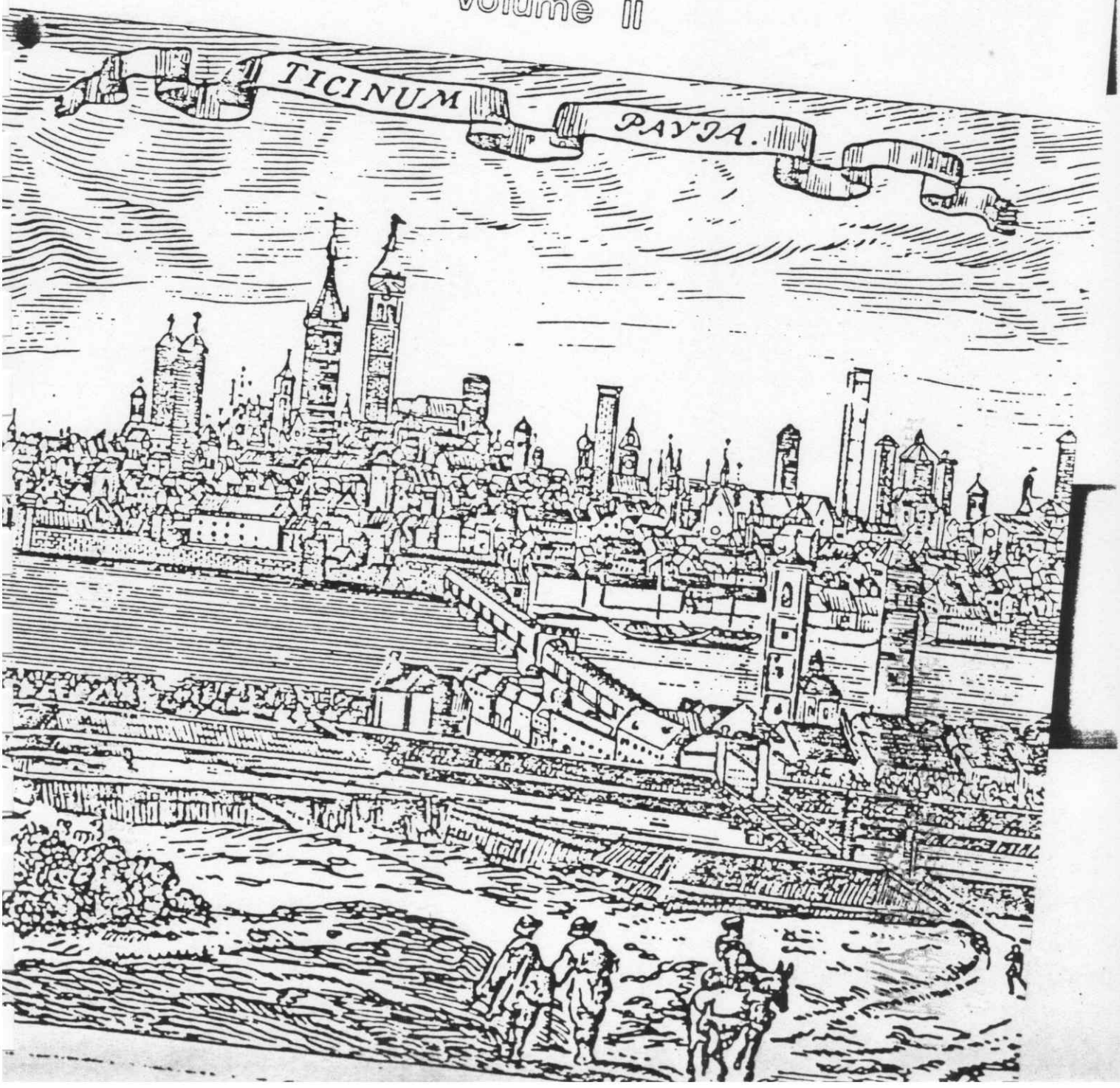
CONGRESSO ANNUALE
ANNUAL CONFERENCE
A.I.C.A.

ASSOCIAZIONE ITALIANA PER IL CALCOLO AUTOMATICO

PAVIA 23-25 Settembre 1981

atti

volume II



IL SISTEMA DI VISIONE DEL ROBOT SUPERSIGMA

R. CASSINIS, L. SCHNICKEL

Progetto di Intelligenza Artificiale del Politecnico di Milano

L'esigenza di avere a disposizione sistemi di visione per robot e' ormai un fatto indiscutibile. Pochi sono tuttavia coloro che, fino ad ora, hanno tentato di realizzare tali dispositivi.

Presso il Laboratorio di Robotica del Politecnico di Milano si sono iniziate da alcuni mesi ricerche e sperimentazioni in tal senso.

L'articolo presenta le caratteristiche dell'architettura usata e del software finora implementato, i primi risultati ottenuti e le problematiche sorte in vista di futuri sviluppi.

1. - INTRODUZIONE.

Di tutti i dispositivi che possono essere collegati ai robot industriali per incrementarne le capacità, i sistemi di visione sono senza dubbio i più utili, ma anche fra i più difficili da realizzare.

Ciò perché i problemi da risolvere sono molteplici, sia dal punto di vista hardware, sia, soprattutto, dal punto di vista software.

Questo articolo illustra il sistema di visione realizzato presso il Laboratorio di Robotica del Politecnico di Milano, con lo scopo di studiare la possibilità di produrre un sistema di buone caratteristiche a un prezzo industrialmente valido.

L'articolo descrive la struttura dei componenti e dei programmi attualmente disponibili sul sistema e i problemi incontrati durante la sua realizzazione.

2. - OBIETTIVI DEL SISTEMA

La necessità di dotare il SUPERSIGMA di un sistema di visione è apparsa evidente fin da quando sono stati scritti i primi programmi sperimentali. Essa deriva principalmente dal fatto che, come tutte le macchine che devono "manipolare" pezzi, il posizionamento iniziale di questi ultimi è fondamentale perché essi possano essere afferrati in maniera corretta. D'altra parte, il costo delle apparecchiature necessarie per il posizionamento è elevatissimo, sia per il prezzo dei componenti, sia perché gli attrezzaggi realizzati in questo modo sono molto poco flessibili, il che vuol dire dover riprogettare e mettere nuovamente a punto buona parte dell'attrezzatura quando si cambia tipo di lavorazione.

La possibilità invece di poter riconoscere visivamente il tipo e la posizione di oggetti posati casualmente sul piano di lavoro risolve molti problemi, con un prezzo comparabile, e in alcuni casi inferiore, a quello delle attrezzature tradizionali.

Il sistema che è stato realizzato è comunque molto flessibile, perché nessuna limitazione "a priori" è stata posta sul software che lo gestisce, e quindi sulle funzioni che esso è in grado di svolgere.

Trattandosi infatti di una apparecchiatura sperimentale, una volta che i programmi relativi al riconoscimento del tipo e della posizione di oggetti saranno a punto, si potrà passare ad obiettivi più ambiziosi, cui si farà cenno più avanti.

E' infine da ricordare che il sistema, che è attualmente montato su uno dei bracci del SUPERSIGMA, può essere adattato a funzionare anche in collaborazione con il robot PUMA, in dotazione al nostro laboratorio, o con qualunque altro sistema di manipolazione automatica.

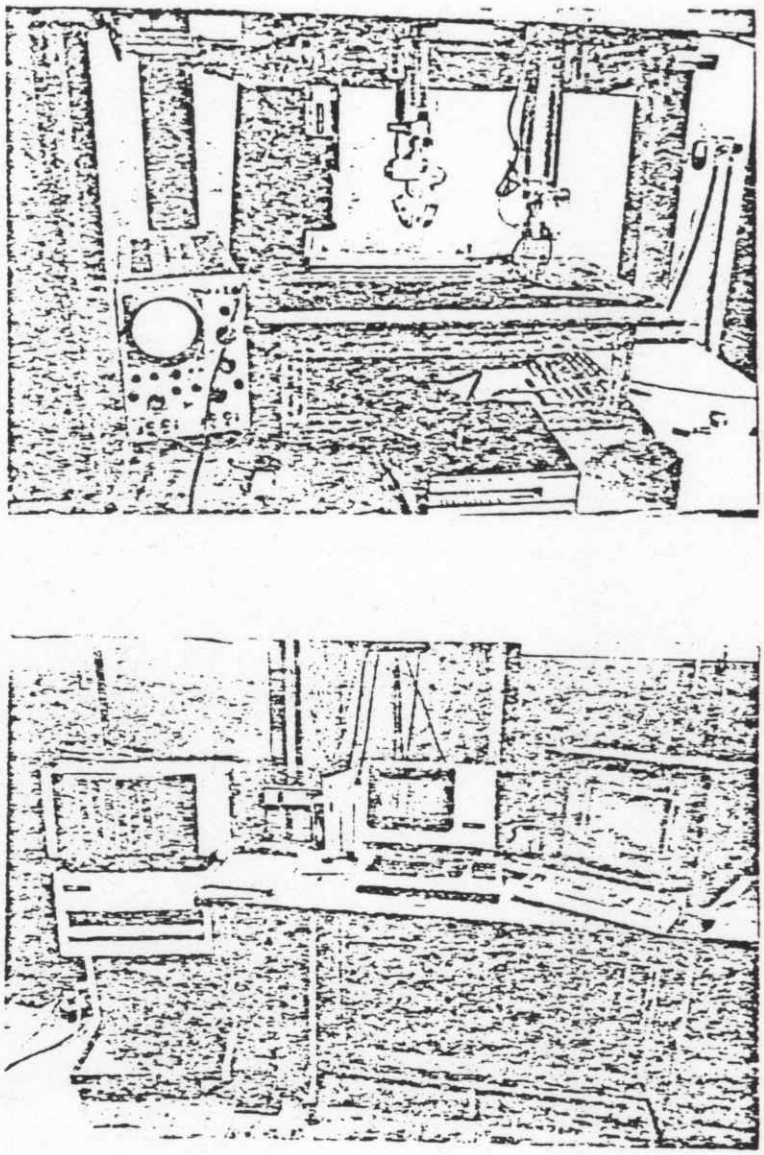


Fig. 1 - Vista globale del sistema.

3. - DESCRIZIONE DEL SISTEMA.

Vediamo ora il tipo di architettura usata presso il Laboratorio di Robotica per lo sviluppo di programmi di visione (Fig. 1).

Il cuore del sistema è una telecamera a stato solido General Electric TN 2200, applicata ad uno dei bracci del SUPERSIGMA, che inquadra la scena da circa 40 cm di altezza, con un campo utile di 15 x 15 cm. Si tratta di una telecamera il cui sensore di immagine è un chip CCD con 128 x 128 elementi fotosensibili. Questo tipo di telecamera, benché ancora poco disponibile sul mercato e quindi relativamente costoso, offre il vantaggio di una grande robustezza e facilità di impiego, soprattutto nei casi in cui sia necessario digitalizzare e acquisire dati a bassa velocità di scansione (cosa impossibile con le telecamere di tipo tradizionale, che richiedono in genere memorie tampone per registrare transitoriamente l'immagine ricevuta).

La telecamera è collegata a un digitalizzatore - controllore che fornisce su 8 bit il valore di luminosità di ciascun pixel e i necessari segnali di sincronismo.

Per ciò che concerne la fase di elaborazione dell'immagine si usa un calcolatore Digital MINC11 con 28 Kbyte di memoria, floppy disks e possibilità di collegamento ad altri sistemi di elaborazione dell'Istituto (PDP11/34 e VAX11/780). La acquisizione dell'immagine avviene attualmente attraverso una porta parallela, gestita a controllo di programma. Questo metodo, che non permette ovviamente alte velocità di trasferimento, porta l'inconveniente che non è possibile riprendere gli oggetti in movimento, e peggiora la qualità dell'immagine, a causa della riduzione della gamma dinamica della telecamera che si verifica alle basse velocità di scansione. E' quindi già stata acquisita, e sarà presto messa in funzione, una interfaccia DMA che consentirà di utilizzare il sistema alla massima velocità consentita dal digitalizzatore.

Una volta acquisita l'immagine, l'elaborazione può iniziare immediatamente, dal momento che tutto il quadro è disponibile in memoria. E' tuttavia anche possibile salvare l'immagine su floppy disk, o trasmetterla agli altri elaboratori a cui il sistema è collegato.

L'apparecchiatura comprende anche un display semi-grafico a colori ISC8001 che permette diversi tipi di rappresentazione dei risultati delle elaborazioni. Anche il terminale del MINC, che è dotato di capacità semi-grafiche, può essere usato allo stesso scopo.

L'illuminazione della scena è realizzata con lampade alogene a luce diffusa o con flash. La scelta del sistema di illuminazione deriva dalla necessità di rendere l'apparecchio il più economico possibile, anche se esistono in commercio sistemi più sofisticati di illuminazione, che permettono di ottenere immagini più contrastate e con meno "disturbi", come ad esempio ombre e riflessi. Inoltre, l'impiego dei flash permette di "fermare" oggetti in movimento, se la scena non è illuminata da altre fonti di luce.

Il collegamento fra il sistema di visione e il controllore del SUPERSIGMA è stato realizzato utilizzando una porta seriale del MINC11, che scambia dati e comandi con un microcalcolatore collegato al MICROBUS del SUPERSIGMA secondo il protocollo di quest'ultimo. Ciò permette di considerare il sistema di visione come tutti gli altri sensori presenti sul robot, e quindi di gestirlo nello stesso modo, con evidenti vantaggi per la semplicità e la funzionalità del software [1].

Uno schema a blocchi dell'apparecchiatura è mostrato in fig. 2.

4. - SOFTWARE DISPONIBILE.

Essendo la ricerca sui sistemi di visione un argomento per noi del tutto nuovo, diverse strade sono state tentate per valutare sia la bontà della nostra architettura, sia la validità di algoritmi originali, sia l'efficienza di metodi e programmi proposti e implementati da altri.

L'obiettivo principale è stato avere a disposizione in un tempo ragionevolmente breve un programma di riconoscimento ed orientamento di pezzi meccanici per dar modo al robot SUPERSIGMA di operare su pezzi posti in posizione casuale sul suo piano di lavoro, in modo analogo ad alcuni sistemi già esistenti [2].

E' stato realizzato un programma [3] che è in grado di estrarre da un'immagine alcune caratteristiche salienti, come ad esempio la misura dell'area del pezzo inquadrato, l'individuazione del centro di gravità, la direzione di assi significativi.

In una prima fase vengono posti sotto la telecamera i pezzi da riconoscere (uno per volta), e di ciascuno vengono estratti i parametri caratteri-

stici che, organizzati e memorizzati su disco, costituiscono la biblioteca degli oggetti conosciuti.

Terminata la fase di apprendimento, ha inizio l'esecuzione del programma di riconoscimento vero e proprio.

Il SUPERSIGMA richiede al sistema di visione di estrarre le caratteristiche dell'oggetto e di compiere una ricerca sulla base dei dati a disposizione per trovare il tipo di pezzo in questione.

Se l'esito è positivo (il pezzo viene cioè riconosciuto), le seguenti informazioni sono inviate al robot:

a) coordinate del baricentro;
b) rotazione dell'oggetto intorno al baricentro, rispetto alla posizione "canonica" memorizzata;
c) giacitura del pezzo (normale o ribaltata).

Le limitazioni di questo programma sono dovute sia all'impossibilità di operare su pezzi che hanno più di due giaciture possibili, sia alla limitatezza dei parametri caratteristici utilizzati.

Inoltre, non è possibile riconoscere pezzi sovrapposti ad altri: perché il programma sia in grado di funzionare, occorre che la telecamera inquadri solo ed esclusivamente il pezzo da riconoscere.

Parallelamente alla implementazione di questo algoritmo, ci si è occupati di un certo numero di programmi sperimentali per il miglioramento delle immagini acquisite nel tentativo di estrarre dalla rappresentazione reale di un oggetto il suo modello.

Di questi fanno parte programmi per la binarizzazione e l'estrazione del contorno dell'immagine.

La telecamera fornisce, come si è detto, il valore di luminosità di ciascun pixel su 8 bit, quindi si ha a disposizione una gamma di 256 livelli di grigio. Questo tipo di informazione è spesso ridondante, e vale talvolta la pena di ricavarne una immagine a due soli livelli di luminosità (bianco o nero). Questa operazione presenta però il problema di trovare il miglior valore per la soglia al di sopra della quale tutti i punti sono considerati bianchi e al di sotto della quale neri (e quindi appartenenti rispettivamente al soggetto o allo sfondo).

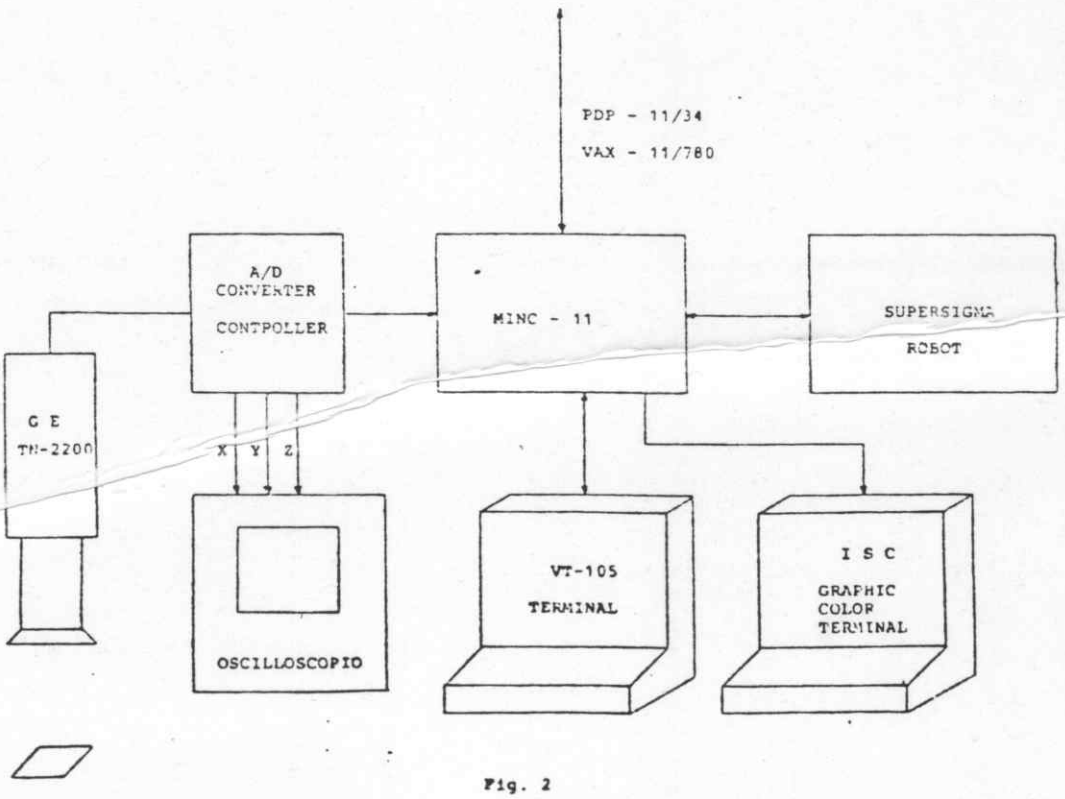


Fig. 2

A causa dell'estrema variabilità delle caratteristiche di illuminazione e contrasto dell'immagine, è impensabile servirsi di un valore fisso per la soglia, ma è necessario usare un metodo per la sua determinazione ottima caso per caso.

Accade inoltre talvolta che le parti più luminose dello sfondo siano più chiare delle parti meno luminose del soggetto. Ciò vuol dire che non esiste un ottimo "globale" per la soglia, ma solo una serie di ottimi "locali".

Per verificare questo fenomeno, è stato scritto un programma che mostra su un display l'istogramma della distribuzione dei diversi livelli di grigio dell'immagine in esame.

In condizioni ottimali, questo istogramma presenta due picchi ben separati, corrispondenti rispettivamente ai livelli di luminosità dello sfondo e del soggetto (Fig. 3).

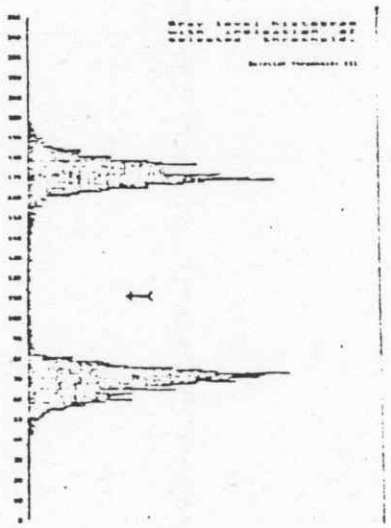


Fig. 3.

In alcuni casi però i due picchi tendono a confondersi, appunto perché i livelli di luminosità dello sfondo e del soggetto sono simili (Fig. 4).

Mediante opportuni comandi, è possibile variare a piacere la soglia di discriminazione fra nero e bianco, e verificare immediatamente il risultato su un altro schermo.

Si può osservare (Fig. 5) che in questo caso non è possibile utilizzare una soglia uguale per tutta l'immagine senza includere parti dello sfondo nel contorno (presunto) del soggetto o, al contrario, senza escludere dall'immagine binaria parti del soggetto stesso.

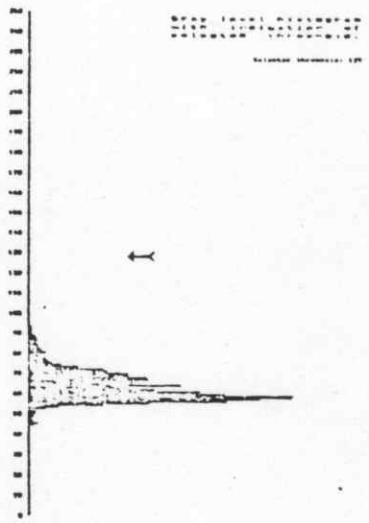


Fig. 4.

E' facile poi constatare che lievi scostamenti dal valore ottimo generano variazioni del perimetro stimato del pezzo (Fig. 6).

E' stato comunque messo a punto un metodo empirico che, sulla base di valutazioni statistiche sulla distribuzione dei valori di luminosità, è in grado, per immagini ben contrastate, di calcolare il valore della soglia migliore.

Per quanto riguarda l'estrazione del contorno del pezzo i primi tentativi sono stati fatti derivando la luminosità dell'immagine secondo una o due direzioni, ottenendo nel secondo caso risultati soddisfacenti, con tempi di elaborazione estremamente contenuti. E' stata però constatata in questo caso la necessità di filtrare l'immagine, a causa dell'eccessiva sensibilità dell'operatore di derivazione al rumore presente.

Infine, è stato ultimamente implementato un algoritmo basato sull'operatore di Huckel per la ricerca del contorno con risultati discreti, ma con tempi di elaborazione assolutamente inaccettabili sul piano pratico.

Si è ora in attesa del risultato di alcuni algoritmi di edge-following che saranno messi a punto nei prossimi mesi.

5. - PROSPETTIVE FUTURE.

Come è stato detto nel paragrafo 2, il sistema hardware non è dedicato a nessuna particolare applicazione. E' quindi possibile pensare a finaliz-

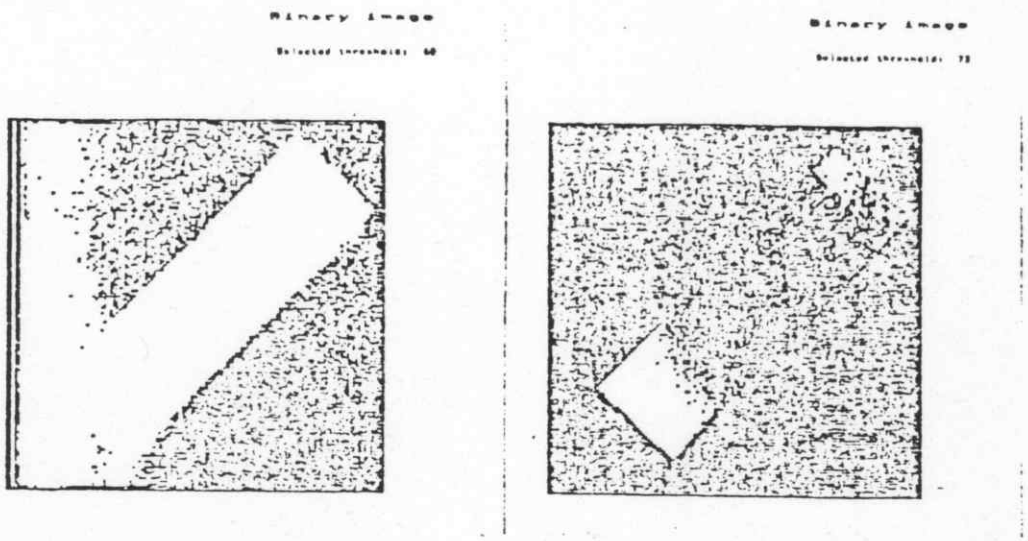

Binary Image
Selected threshold: 60
Binary Image
Selected threshold: 72

Fig. 5.

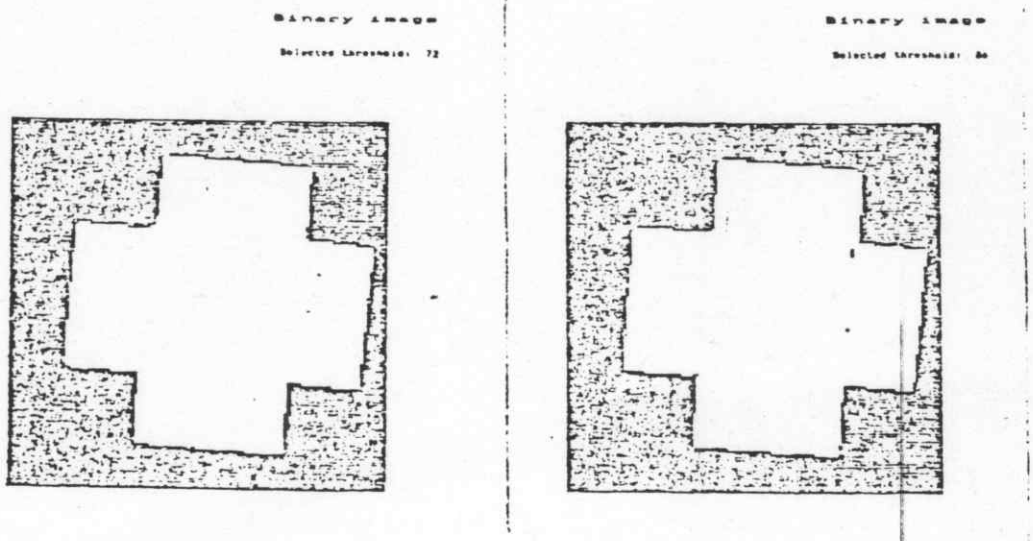

Binary Image
Selected threshold: 72
Binary Image
Selected threshold: 86

Fig. 6.

zazioni diverse da quelle attualmente allo studio.

E' tuttavia molto importante tenere presente che, allo stato attuale della tecnologia, alcuni modi di funzionamento dei sistemi di visione sono ancora impossibili. Un'analisi dettagliata di questo problema si può trovare in [4]; basterà qui ricordare che, finché i dati richiesti sono pochi (ad esempio, tipo di oggetto visto e coordinate che ne definiscono la posizione), non esistono in via teorica ostacoli al collegamento fra visione e sistema di controllo del robot: quando invece occorre una interazione più stretta (ad esempio, correzione automatica dei movimenti del robot in funzione di osservazioni visive), il collegamento non è possibile, tranne che in casi particolari, anche ammesso che il sistema di visione in se stesso sia costruibile.

Si prevede quindi che il lavoro dei prossimi anni sarà sì dedicato al raffinamento degli strumenti attualmente esistenti, ma che occorrerà occuparsi profondamente del problema della "comprensione delle scene", che è per un certo verso simile a quello della comprensione dei linguaggi naturali, e dei problemi di generazione automatica dei piani.

Un altro argomento interessante, di cui stiamo incominciando ad occuparci, è la possibilità di parallelizzazione degli algoritmi impiegati, per eseguirli su reti di microcalcolatori invece che su un mini, come si fa attualmente. Questa filosofia consentirebbe di adeguare la velocità degli algoritmi usati a quella, necessariamente elevata, richiesta da molti processi industriali.

6. - CONCLUSIONI.

Il sistema illustrato non presenta certamente particolari caratteristiche di originalità, almeno per quanto riguarda la parte hardware: esso ha comunque già fornito buoni risultati, non ostante sia in funzione da pochi mesi.

Si ritiene comunque che, nel giro di meno di un anno, esso potrà essere impiegato utilmente in diversi processi di manipolazione e di assemblaggio.

7. - BIBLIOGRAFIA.

[1] R. Cassinis, An Example of a Distributed Intelligence Discrete Process Controller, Digital Systems for Industrial Automation, Vol. 1 - N. 1 - 1981.

[2] G. J. Gleason, G. J. Agin, A Modular Vision System for Sensor-Controlled Manipulation and Inspection, Proc. 9th I.S.I.R., Washington, 1979.

[3] M. Somalvico, C. Viviani, Algoritmo di riconoscimento dell'orientamento di pezzi piani, Atti Congresso AICA 1980.

[4] R. Cassinis, La sensorialità nei robot: problemi e prospettive, Automazione e Strumentazione, Vol. XXIX - N. 4 - Aprile 1981.

STATO ATTUALE

E PROSPETTIVE DELLA ROBOTICA

IN ITALIA

=*=*=*=*=*=*=*=*=*=*=*=

ANALISI DI CONNETTIVITA'

ED ESTRAZIONE DI CARATTERISTICHE

IN ITALIA

ing. Paolo Alberti
ing. Renato Marchi

Milano, 4-5 marzo 1982

=*=*=*=*=

Sommario

I sistemi di visione per la Robotica Industriale sono oggi impiegati in molti compiti di ispezione e riconoscimento in cui si richiedono, fra l' altro, tempi di elaborazione brevi per non rallentare i processi di produzione.
Il sistema presentato in questa memoria effettua l' analisi di connettività e l' estrazione delle caratteristiche salienti dell' immagine. Esso utilizza un algoritmo di analisi sequenziale, effettuata riga per riga, che permette di ricavare le caratteristiche desiderate durante una singola scansione dell' immagine. Il sistema lavora su immagini discretizzate, provenienti da una telecamera allo stato solido, dopo averle binarizzate e compresse utilizzando la codifica Run-Length Coding. L' algoritmo è facilmente parallelizzabile, e permette quindi la sua implementazione su macchine ad intelligenza distribuita; inoltre non si pongono limiti alla complessità dell' immagine. Le caratteristiche disponibili al termine dell' elaborazione permettono il riconoscimento degli oggetti inquadrati dalla telecamera, del loro orientamento, del loro eventuale ribaltamento e dei loro rapporti d' inclusione sulla base di dati campione precedentemente memorizzati. Il sistema è scritto in OMSI-PASCAL e quindi facilmente portabile.

1. INTRODUZIONE

L' attuale livello di automazione raggiunto dall' industria manifatturiera sta imponendo la necessità di utilizzare macchine in grado di rilevare, in modo veloce e sufficientemente accurato, la situazione del campo [6]. I sistemi di visione stannno quindi assumendo una grande importanza sia nell' ambito della manipolazione e dell' assemblaggio mediante robots, che in quello dell' ispezione e del controllo [4].
L' analisi automatica di immagini comporta l' elaborazione di grosse quantità di dati provenienti da sensori elettro-ottici (telecamere). L' implementazione di algoritmi che utilizzano tutte le informazioni disponibili porta a risultati inaccettabili, dal punto di vista industriale, a causa del tempo di elaborazione troppo elevato. Inoltre, anche diminuendo la mole dei dati con la binarizzazione dell' immagine, una risoluzione tradizionale del problema [8], ottenuta con un' analisi punto per punto della matrice che rappresenta il quadro visivo, non porta a buoni risultati sia per i tampi lunghi di esecuzione che per la necessità di calcolatori di grosse dimensioni.

Nella presente memoria viene presentato un sistema sperimentale per l' elaborazione di immagini realizzato con i seguenti obiettivi:

- ottenere tempi di elaborazione accettabili dal punto di vista industriale
- calcolare un insieme di caratteristiche sufficiente ad identificare degli oggetti e determinare la loro posizione, l' orientamento e la giacitura
- poter analizzare immagini comunque complesse (oggetti contenuti in fori di altri)
- poter facilmente parallelizzare le operazioni su macchine multi-processore

Questo ha comportato la decisione di lavorare su immagini binarizzate (bianco/nero) codificate in un formato adatto ad una veloce analisi di connettività.

La scelta del linguaggio di programmazione PASCAL è dovuta a considerazioni che riguardano sia la portabilità del prodotto su macchine diverse da quella su cui il sistema è stato sviluppato che la "autodocumentazione" del codice, utile nella fase di sviluppo a cui collaborano più progettisti.

Il sistema a cui si riferisce questa memoria esegue una elaborazione di immagini discretizzate, provenienti da una telecamera a stato solido, su cui vengono effettuate le seguenti operazioni:

- BINARIZZAZIONE
- RUN-LENGTH CODING
- ANALISI DI CONNETTIVITA'
- ESTRAZIONE DI CARATTERISTICHE

Nell' attuale realizzazione (Vers. 3.0) il programma è stato segmentato per poter disporre di memoria centrale sufficiente ad analizzare immagini molto complesse, espandere il sistema con l' aggiunta di nuovi segmenti e memorizzare i dati campione.

2. HARDWARE

L' HARDWARE utilizzato consiste di:

- Telecamera allo stato solido: GE TN-2200, che discretizza l'immagine in 128x128 elementi ognuno dei quali e' caratterizzato da un livello di grigio compreso fra 0 e 255
- Interfaccia in DMA per l'acquisizione dell'immagine
- Elaboratore DEC MINC-11 con memoria da 28K-Word
- Terminale video DEC VT 105 per l'interazione con l'utente e la visualizzazione dei risultati
- Monitor a colori ISC 8001 per la visualizzazione grafica dei risultati

Si prevede di sostituire il MINC-11 con un nodo MODIAC, appena questo sarà disponibile.

3. DESCRIZIONE DEL SOFTWARE

3.1. BINARIZZAZIONE

Viene effettuata in base ad una soglia S [5] specificata interattivamente dall' utente, 0<= S <=255. Lo scopo è di ridurre l'informazione da trattare e quindi rendere più veloce l'operazione d' analisi. Ad ogni pixel della matrice che rappresenta l' immagine è associato un valore VAL (0<= VAL <=255) che indica il livello di grigio. La binarizzazione consiste nel sostituire a VAL un valore VALBIN, appartente all' insieme {0,1}, in base al risultato del confronto con la soglia :

```
if VAL < S     then VALBIN:='0'
               else VALBIN:='1'
```

Attualmente la convenzione adottata assegna '0' allo sfondo (scuro) e '1' all' oggetto (chiaro).

3.2. RUN-LENGTH CODING

Un' immagine binaria è generalmente rappresentata da una matrice di NxN elementi (detti PIXELS) che possono assumere due soli valori (indicati con 0 e 1). La struttura dati che può rappresentarla è quindi una matrice bidimensionale che occupa parecchia memoria, visto che in genere N è dell'ordine delle centinaia (nel nostro caso N=128). Inoltre, gli algoritmi che operano direttamente sulle matrici analizzandone i singoli elementi non sono adatti a causa della loro scarsa efficienza dovuta al numero elevato di pixels da considerare. La codifica "run-length" (RLC) viene incontro a queste necessità in quanto opera una compressione dell' immagine creandone una rappresentazione tabellare efficientemente utilizzabile da algoritmi di analisi di figure piane . La compressione operata dalla RLC avviene senza perdita di informazione, si può cioè risalire alla rappresentazione originale [3]. La codifica fornisce, per ogni riga della matrice originale, solo i punti di transizione da 0 a 1 e viceversa. Più esattamente si segnalano i punti appartenenti

all' oggetto che hanno come punto adiacente sinistro un punto dello sfondo, e i punti dello sfondo che hanno come adiacente sinistro un punto dell' oggetto. La scelta dell' identificatore di sfondo e di oggetto è parametrizzata. Ogni riga della matrice binarizzata viene scandita segnalando quando avviene un cambiamento di valore dei pixels. La sequenza delle ascisse dei punti in cui si è verificato il cambiamento viene memorizzata nei campi X1 e X2 dei records della tabella delle transizioni.

3.3. ANALISI DI CONNETTIVITA'

3.3.1. Note sull' argomento.

Lo scopo è di individuare ed etichettare tutte le componenti connesse dell' immagine (blobs).
L'analisi viene svolta sulla tabella delle transizioni, ottenuta dalla RLC, prendendo in considerazione solo due righe consecutive alla volta:

- la riga corrente (indicata con k)
- la riga precedente (indicata con h=k-1)

Questa scelta permette, se necessario, di risparmiare memoria e di parallelizzare le operazioni di acquisizione ed elaborazione o di elaborazione di parti diverse della tabella contamporaneamente. La codifica matrice/tabella, come del resto la binarizzazione, può essere facilmente realizzata hardware velocizzando quindi ulteriormente il sistema di visione.
Per analisi di connettività si intende l'individuazione di tutte le superfici connesse formate da elementi (PIXELS) dello stesso valore. Di ogni componente connessa individuata, il sistema calcola le caratteristiche principali e le relazioni di inclusione con le altre componenti. Una generica componente connessa dell' immagine è detta BLOB ed ha come prima caratterizzazione il valore dei pixels che le appartengono. Nel nostro caso di immagine binaria i pixels possono assumere solo due valori, genericamente indicati con 0 e 1, quindi anche i blobs possono essere di due soli tipi. Nella tabella dei risultati (fig. 3) sono indicati come BODY (regione indicante un corpo) e HOLE (regione indicante un foro o lo sfondo).
In genere, data una immagine, saranno individuati più BLOBS distinti di tipo body o hole, che indicheranno, mediante le relazioni di inclusione, uno di questi casi:

- corpi distinti
- corpi con fori
- corpi contenenti altri corpi nei fori

o una loro composizione.

NOTAZIONI UTILIZZATE.

Una riga della tabella delle transizioni è formata da una sequenza finita di numeri interi, che rappresentano i punti in cui si verifica una variazione del valore dei pixels, che ha questo aspetto:
...X1(i-1,k), X2(i-1,k), X1(i,k), X2(i,k), X1(i+1,k), X2(i+1,k)...
dove X1 e X2 rappresentano gli estremi dell' i-esimo segmento della k-esima riga. Inoltre:

- X1: punto appartenente alla superficie di un corpo ed il cui punto adiacente sinistro giaccia sullo sfondo.
- X2: punto appartenente allo sfondo il cui punto adiacente sinistro appartenga alla regione di un corpo.

Una riga con più segmenti sarà quindi rappresentata da una sequenza X1,X2,X1,X2...
Si dice COPPIA di transizione una coppia X1,X2 rappresentante un segmento. Dato un segmento:

```
            ------------
            X1          X2
```

X1 rappresenta il suo estremo sinistro e X2 il destro. Si parla però di confine (bordo) tipo 1 riferendosi ad X1 e di confine (bordo) tipo 2 riferendosi ad X2. Ciò perchè gli attributi sinistro e destro vengono riservati ai confini dei blobs: mentre per un corpo il confine tipo 1 (X1) è il confine sinistro, per un foro il confine tipo 1 (X1) è il destro. Viceversa per X2.

In corrispondenza di X1 e X2 esistono 01 e 02, che rappresentano l' etichetta dei blobs a cui i bordi X1 e X2 appartengono. Le sequenze di X1 X2 sono i dati da elaborare, 01 e 02 sono aggiornati durante l' elaborazione (all' inizio dell' elaborazione sono indefiniti).

3.3.2. Descrizione dell' algoritmo.

L' algoritmo è organizzato in modo che elabori tutte le informazioni relative ad una riga prima di passare alla successiva.
L' elaborazione di una riga consiste in:

- leggere una coppia X1(i,k), X2(i,k)
- verificare a quale blob appartengono questi confini mediante confronti con la riga precedente che valutano la continuazione dei bordi
- inserire nei rispettivi 01(i,k), 02(i,k) l' etichetta del blob di appartenenza e aggiornamento delle sue caratteristiche

In caso di continuazione, ad es. di un bordo tipo 1, considerando i e j gli indici di coppia corrente nelle righe k e h, si ha:

$$01(i,k) := 02(j,h)$$

e quindi l' assegnazione a 01(i,k) è immediata.

In caso di non continuazione si deve valutare invece se l' anomalia è dovuta alla linea h e, in questo caso, viene fatto variare l' indice J e rioperato il confronto, oppure alla linea k, ed in questo caso l' etichetta da assegnare deve essere "nuova" (si è in presenza di creazione), e si incrementa l' indice i. Si vede quindi che gli indici i e j, che scorrono le coppie rispettivamente nelle righe corrente e precedente, possono essere diversi.

Compito principale delle strutture di controllo del modulo di analisi di connettività è quindi di coordinare la scansione delle righe con le operazioni di accumulo delle caratteristiche parziali. Questo algoritmo di analisi di connettività non pone limiti alla complessità degli oggetti o delle immagini da analizzare, cioè non vi sono limitazioni al numero di corpi,

di fori presenti in ogni corpo ed ai livelli di inclusione di un corpo dentro un altro.

3.3.3. Considerazioni sulle condizioni di continuazione.

Dati due segmenti:

X1 (j,h), X2 (j,h) linea precedente

X1 (i,k), X2 (i,k) linea corrente

con h=k-1, le condizioni generali affinchè il confine 1 X1(j,h) abbia una continuazione in X1(i,k) sono [2]:

$$X2(i-1,k) < X1(j,h) < X2(i,k) \qquad (1)$$

$$X2(j-1,h) < X1(i,k) < X2(j,h) \qquad (2)$$

e per il confine 2 si hanno le seguenti:

$$X1(i,k) < X2(j,h) < X1(i+1,k) \qquad (3)$$

$$X1(j,h) < X2(i,k) < X1(j+1,h) \qquad (4)$$

Scomponendo le precedenti condizioni in test elementari si ottengono:

$$X2(i-1,k) < X1(j,h) \qquad (1.A)$$

$$X1(j,h) < X2(i,k) \qquad (1.B)$$

$$X2(j-1,h) < X1(i,k) \qquad (2.A)$$

$$X1(i,k) < X2(j,h) \qquad (2.B)$$

$$X1(i,k) < X2(j,h) \qquad (3.A)$$

$$X2(j,h) < X1(i+1,k) \qquad (3.B)$$

$$X1(j,h) < X2(i,k) \qquad (4.A)$$

$$X2(i,k) < X1(j+1,h) \qquad (4.B)$$

Si nota che 1.B=4.A e 2.B=3.A .
Rimangono quindi sei diverse relazioni da applicare per le verifiche di continuazione, e su queste ci sono interessanti considerazioni da fare: Osservando la 1.A e la 4.B si nota che coincidono a meno degli indici di coppia che hanno differenza unitaria. Ciò implica che se le condizioni di continuazione sono applicate in sequenza, valutare prima la 1.A e poi, una volta incrementati gli indici, valutare la 4.B significa valutare la stessa relazione. Considerando il fatto che una continuazione di confine tipo 1 viene esaminata dopo una avvenuta continuazione di un confine tipo 2 (oppure come prima coppia: i=j=1), si vede come sia inutile tenere in considerazione entrambi i test 1.A e 4.B . Manteniamo 4.B ed eliminiamo 1.A . Lo stesso ragionamento vale per i test 2.A e 3.B . Manteniamo 3.B ed eliminiamo 2.A .
Rimangono quindi i seguenti quattro test:

$$X1(j,h) < X2(i,k) \qquad (1.B)$$

$$X1(i,k) < X2(j,h) \qquad (2.B)$$

$$X2(j,h) < X1(i+1,k) \qquad (3.B)$$

$X2(i,k) < X1(j+1,h)$ (4.B)

che, applicati in sequenza nella scansione di una riga, sono sufficienti alla rilevazione di qualsiasi situazione.

Più precisamente si possono verificare le seguenti situazioni:

a) 1.B=true e 2.B=true : CONTINUAZIONE BORDO 1

b) 3.B=true e 4.B=true : CONTINUAZIONE BORDO 2

c) 1.B=false ---> $X1(j,h) > X2(i,k)$:

CREAZIONE DI UN CORPO

d) 2.B=false ---> $X1(i,k) > X2(j,h)$: due sottocasi

TERMINAZIONE DI UN CORPO
ASSORBIMENTO DI UN FORO

e) 3.B=false ---> $X2(j,h) > X1(i+1,k)$:

CREAZIONE DI UN FORO

f) 4.B=false ---> $X2(i,k) > X1(j+1,h)$: due sottocasi

TERMINAZIONE DI UN FORO
ASSORBIMENTO DI UN CORPO

Si noti che non può esistere una situazione 1.B=2.B=false oppure 3.B=4.B=false.
Nel caso di una sola h-coppia e di una sola k-coppia è sufficiente valutare i test 1.B e 2.B per garantire la continuazione di entrambi i bordi. Se una delle due condizioni non è verificata , indipendentemente se una o l'altra, si può affermare che sia verificata la terminazione di un corpo in riga h e la creazione di uno in riga k. Nel caso di creazione il bordo destro del blob neonato è la transizione seguente a quella del bordo sinistro. Nel caso di terminazione i bordi sono rappresentati da due transizioni adiacenti.

3.4. ESTRAZIONE DI CARATTERISTICHE

Ad ogni blob è associato un insieme di dati detto "descrittore del blob". Questo descrittore viene usato per memorizzare i risultati parziali e finali ottenuti dall'analisi dell' immagine. Campi tipici di questi descrittori sono ad esempio AREA, PERIMETRO, ecc. I descrittori non sono uguali per tutti i blobs: una classe di caratteristiche è comune mentre per i blobs di tipo body selezionati in base ad un' area minima, detti KING, viene calcolato un insieme aggiuntivo di caratteristiche. Descriviamo ora le caratteristiche calcolate indicandole con i nomi che compaiono sul tabulato dei risultati (vedi figg. 2 e 3).

INSIEME COMUNE:

- NOME: etichetta che identifica il Blob in esame. L'etichetta '@' rappresenta lo sfondo, concettualmente sempre presente.

FIG. 1

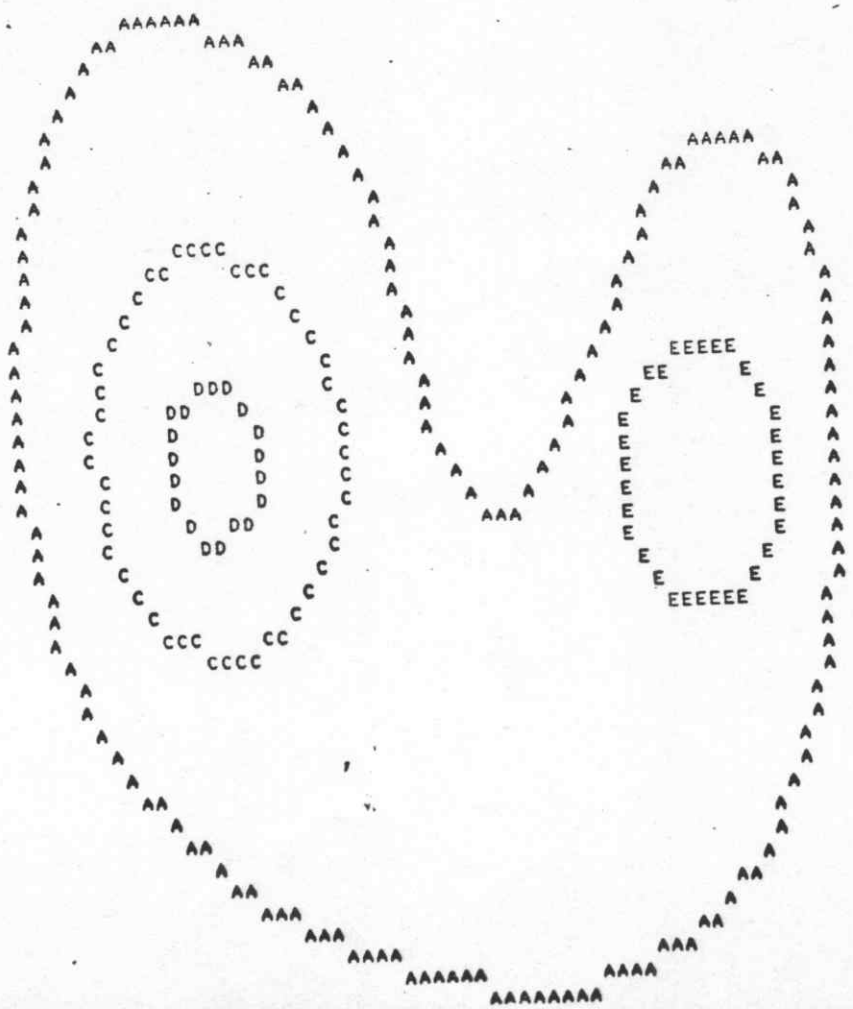


FIG. 2

NOME	TIPO	STATO	GENITORE	PERIMETRO	AREA	MOMX	MOMY	XBAR	YBAR	DMIN	DMAX	FIGLI
a	----	----	-	----	---	-----	-----	---	---	---	---	A
A	BODY	TERMINATO	a	238.049	1474	108067	37598	73	26	4	32	E, C
B	BODY	ASSORBITO	a	43.271	256	23877	4319	0	0	0	0	E, C
C	HOLE	TERMINATO	A	53.424	172	10004	3595	58	21	0	0	D
D	BODY	TERMINATO	C	23.543	44	2548	940	58	21	3	4	---
E	HOLE	TERMINATO	A	33.136	87	8050	1962	93	23	0	0	---

RMIN: 2 RMAX: 46 CMIN: 44 CMAX: 103

SELEZIONANDO I BODY IN BASE AD UN'AREA MINIMA
DI 6 PIXELS, SONO INDIVIDUATI
I SEGUENTI 2 CORPI PRINCIPALI (KING)
DI CUI SI CALCOLANO CARATTERISTICHE AGGIUNTIVE.

KING	NUMF	ATF	AMF	COMPAT	EFV	DBV	EFL	DBFL	DBFVL	DBFM	ANGPV	ANGPL	INTANG	FP	XPMIN	YPMIN	XPMAX	YPMAX
A	2	259	129.5	38.44	C	15.8	E	20.2	35.1	18.0	5.498	3.773	-1.725	-1	76	23	47	7
D	0	0	0.0	12.60	-	0.0	-	0.0	0.0	0.0	5.498	1.816	-3.682	-1	60	19	57	25

KING	ANGBVIC	ANGBLON	INTBFANG	FF	FM
A	3.463	6.134	2.671	1	0
D	0.000	0.000	0.000	0	0

*** TEMPI DI ESECUZIONE ***

PROCEDURA	MIN	SEC	CENT
BINARIZZA	0	4	13.32
RUN-LENGTH CODING	0	2	40.03
SCANCOTAB	0	2	68.35
CORREGGIRISULTATI	0	0	0.00
BARICENTRI	0	0	1.67
FORIFEATURES	0	0	1.63
DISTBAR	0	0	21.67
TEMPO TOTALE	0	9	46.67

Fig. 3

- TIPO: distingue i blobs in due tipi, body e hole. Con le attuali convenzioni body corrisponde al corpo effettivo sotto la telecamera (con sfondo nero).
- STATO: indica le etichette significative (terminate).
- GENITORE: etichetta del blob che contiene, al suo interno, il blob in esame.
- PERIMETRO: è accumulato durante la scansione utilizzando le regole di calcolo della distanza tra due punti.
- AREA: è accumulata durante la scansione sommando il numero dei pixels.
- MOMX, MOMY: momenti rispetto ai due assi(vedi par. 4)
- XBAR, YBAR: coordinate del baricentro.
- DMIN, DMAX: distanza minima e massima del baricentro dal perimetro.
- LISTA DEI FIGLI: fornisce, per ogni blob, le etichette di quelli in esso contenuti.

INSIEME AGGIUNTIVO. (per blobs di tipo=body e area > areamin)

- NUMF: numero dei fori.
- ATF: area totale dei fori.
- AMF: area media dei fori.
- EFV: etichetta del foro più vicino al baricentro del KING in esame.
- DBV: distanza fra i baricentri del KING e del foro più vicino.
- EFL: etichetta del foro più lontano dal baricentro del KING in esame.
- DBFL: distanza fra il baricentro del KING e il baricentro del foro più lontano.
- DBFVL: distanza fra i baricentri dei fori più vicino e più lontano (dal baricentro del king).
- DBFM: distanza media dei fori dal baricentro del KING.
- ANGPV: angolo formato dalla retta passante per il baricentro del KING e il punto più vicino del perimetro con l' asse X (riga).
- ANGPL: angolo formato dalla retta passante per il baricentro del KING e il punto più lontano del perimetro con l'asse X (riga).
- INTANG: differenza fra i due angoli precedenti che permette, in certi casi , di calcolare la giacitura (FP).
- XPMIN,YPMIN,XPMAX,YPMAX: coordinate dei punti più vicino e più lontano del perimetro del KING dal suo baricentro.
- ANGBVIC: angolo formato dalla retta passante per i baricentri del KING e del foro più vicino con l' asse X.
- ANGBLON: angolo formato dalla retta passante per i baricentri del KING e del foro più lontano con l' asse X.
- INTBFANG: differenza fra i due angoli precedenti che permette, in certi casi, di valutare il ribaltamento (FF).

- FM: terzo metodo, dopo FP e FF, per la valutazione della giacitura. Viene utilizzato quando il baricentro del corpo giace sul perimetro ed esiste un solo foro per cui non sono utilizzabili gli altri due metodi. Se anche FM non è utilizzabile, per mancanza di fori, non è possibile valutare la giacitura con questi metodi.
- BOUNDING BOX: fornisce le righe (RMIN, RMAX) e le colonne (CMIN, CMAX) che racchiudono gli oggetti.
- LISTE PERIMETRI: per ogni blob terminato è disponibile una lista doppiamente concatenata i cui nodi contengono le coordinate dei punti del suo contorno.

Inoltre si possono calcolare delle combinazioni algebriche delle caratteristiche prima indicate. Un esempio è:

- COMPATTEZZA: (perimetro)**2 / area.

Gli angoli sono tutti espressi in radianti.

4. ACCUMULO SEQUENZIALE DEI MOMENTI

I momenti MOMX e MOMY hanno il seguente significato:

$$MOMX = \sum_{i,j=1}^{N} b_{ij}\, x_{ij} \quad , \qquad MOMY = \sum_{i,j=1}^{N} b_{ij}\, y_{ij}$$

dove:

- $b_{ij} \in \{0,1\}$ pesa il fatto che il pixel in esame possa appartenere allo sfondo (b=0) o all' oggetto (b=1)
- x : coordinata X del pixel (i,j)
- y : coordinata Y del pixel (i,j)

In un approccio sequenziale all' analisi di connettività, sorge il problema dell' accumulo dei momenti (di primo e secondo ordine). Si vuole evitare di riscandire l' immagine per effettuare tale operazione e si vuole, inoltre, trattare segmenti di immagine anzichè singoli pixels. Indicando con X le colonne e Y le righe, si vede che l' accumulo di MOMY non crea problemi perchè le distanze dall' asse X sono uguali per tutti gli elementi del segmento, mentre nell' accumulo di MOMX ogni pixel ha distanza diversa. Invece di usare formule approssimate, che coinvolgono operazioni aritmetiche complesse si è preferito adottare la soluzione seguente, in base anche alla considerazione che il sistema completo di visione funzionerà in modo ciclico.
Dato un segmento X1,X2

```
            ------------
           X1           X2
```

indichiamo con :

- X1,X2 : valori dei punti di transizione (vedi Par. 3.3.1)
- A= X2-X1-1 (0 <= A <= N-1)

Il MOMX si può esprimere come :

MOMX= X1+(X1+1)+(X1+2)+...+(X1+A)= (X2-X1)*X1+(1+2+...+A). Si noti che X1 e X2 sono informazioni note mentre la somma 1+2+...+A sarebbe da calcolare ogni volta in dipendenza del valore di A. I valori delle somme sono memorizzati in un vettore di N elementi interi in modo da poterli indirizzare con la differenza (X2-X1). Ciò comporta che le somme vengano calcolate una volta sola all' inizializzazione del sistema.

Questa soluzione si estende facilmente anche al calcolo dei momenti del secondo ordine.

5. OSSERVAZIONI

La memoria occupata dal codice rilocabile è di circa 27K-words(16 bit). E' stata effettuata una suddivisione in quindici segmenti ottenendo un' occupazione complessiva delle regioni di overlay di 14K-words.

Nelle figg. 2 e 3 è riportato un esempio dei risultati ottenuti analizzando l' immagine di fig. 1. I tempi di esecuzione vengono riportati solo per valutare il peso dei diversi moduli del sistema. Infatti i valori dei tempi in assoluto sono penalizzati dal fatto che il sistema è segmentato e lavora con i floppy-disks, l' elaboratore non ha caratteristiche di velocità adatte a queste applicazioni e il sistema presentato costituisce una soluzione del tutto generale al problema. Esso calcola una serie di caratteristiche non strettamente necessarie a qualificare l' immagine. Si puo notare che i moduli di binarizzazione e di codifica (RUN LENGTH-CODING) impiegano una parte rilevante del tempo di calcolo totale. Questi due moduli sono facilmente realizzabili hardware ottenendo così tempi inferiori di un ordine di grandezza.
SCANCOTAB si riferisce all' analisi di connettività e all' accumulo delle caratteristiche parziali.
CORREGGIRISULTATI diventa significativa al crescere del numero di oggetti e dei livelli di inclusione.
BARICENTRI, FORIFEATURES, DISTBAR si riferiscono al tempo impiegato per calcolare i baricentri, le caratteristiche dei fori e le distanze fra i baricentri. I risultati finali sono significativi solo per le etichette con stato TERMINATO.
La visualizzazione dei risultati sul monitor grafico a colori (vedi ad es. foto 1 e 2) è utile all' operatore, specialmente nella fase di costruzione dell' archivio contenente i dati campione, per valutare velocemente il corretto funzionamento del sistema .

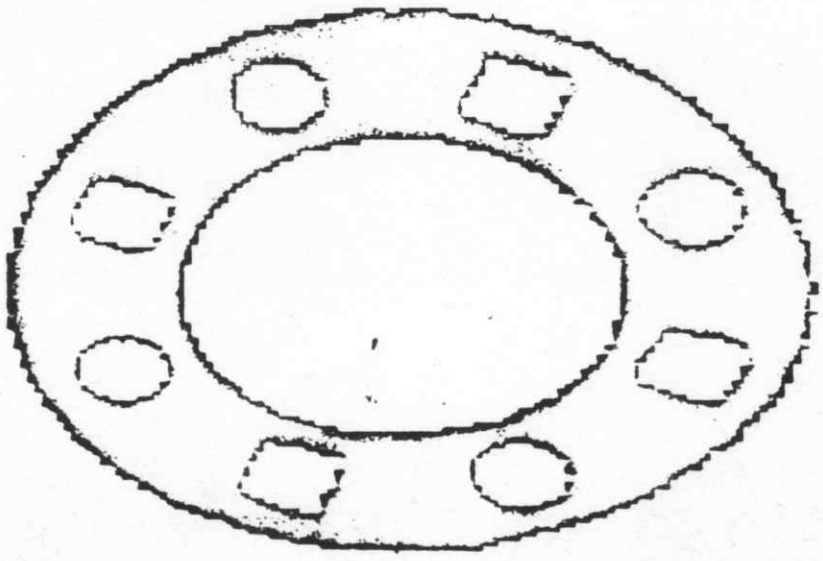


Foto 1: visualizzazione dell' immagine binarizzata.

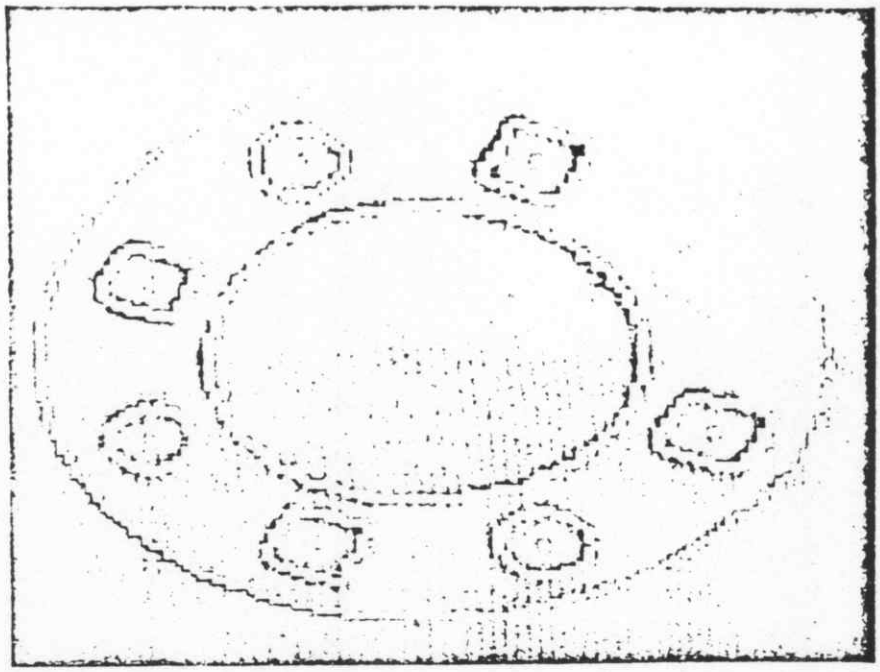


Foto 2: visualizzazione dei contorni dei blobs e dei loro baricentri.

6. BIBLIOGRAFIA

1. Agin G.J., Duda R.O. - SRI vision research for advanced industrial automation - atti II USA-Japan Computer Conference, Tokyo, August 1975.
2. Agrawala A.K., Kulkarni A.V. - A sequential approach to the extraction of shape features - Computer Graphics and Image Processing 6, pp. 538-557 (1977).
3. Di Zenzo S. - Problemi di compressione delle immagini nelle basi di dati geografiche - atti AICA, Bologna 1980.
4. Gini M., Stucchi E. - Analisi di immagini video per l' industria manifatturiera - Automazione e Strumentazione, n. 4, Aprile 1981.
5. Gleason G.J., Agin G.J. - A modular vision sistem for sensor-controlled manipulation and inspection - atti IX International Symposium on Industrial Robots, Washington, USA, 1979.
6. Nicolò V. et altri - Sistemi di visione in produzione - Notiziario Tecnico AMMA, n. 10, 1981.
7. Rosen C.A. - Machine vision and Robotics: Industrial requirements - In Computer Vision and sensor-based Robots, Plenum Press, New York, 1979.
8. Rosenfeld A., Pfaltz J.L. - Sequential operations in digital picture processing - JACM, Vol. 14, n. 4, pp. 471-494, (Ottobre 1966).

STATO ATTUALE

E PROSPETTIVE DELLA ROBOTICA

IN ITALIA

=*=*=*=*=*=*=*=*=*=*=

UN SISTEMA OPERATIVO

CON MACCHINE DI VISIONE

prof. Marco Somalvico
sig. M. Bossi
sig. Flavio Maggioni
sig. R. Sempio

Milano, 4-5 marzo 1982

=*=*=*=*=

SOMMARIO

Il lavoro illustra i criteri di progetto e le prime attività sperimentali per la realizzazione delle primitive funzionali e del sistema operativo di una macchina di visione.

Il sistema sperimentale è attualmente funzionante sul calcolatore MINC-11 del Laboratorio di Robotica del Progetto di Intelligenza Artificiale del Politecnico di Milano

1. INTRODUZIONE

I manipolatori automatici, più o meno intelligenti, hanno ormai raggiunto un alto grado di sviluppo tecnologico ed una notevole diffusione, soprattutto nei campi industriale e della ricerca [1]. Ovvia quindi la necessità di dotarli di un apparato visivo che li metta in grado di avvicinarsi sempre di più al modello di comportamento umano [2].

Una telecamera, la sua interfaccia, ed un calcolatore opportunamente programmato, ad essa collegato, formano gli organi per la visione di un robot [3].

Tale apparato, unito a quello sensoriale ed alle capacità motorie del manipolatore, permette di svolgere un grande numero di attività che, altrimenti, sarebbero per l'uomo di difficile realizzazione e/o pericolose e/o banalmente ripetitive. Si pensi ad esempio al controllo di produzione di un pezzo dotato di un grande numero di piccoli fori, oppure ad operazioni di assemblaggio in una catena di montaggio [10].

I principali campi di applicazione di un sistema di visione, detto anche macchina di visione, sono quelli dell'analisi ed elaborazione di immagini, che sono una premessa necessaria per permettere degli stretti controlli di produzione di un'industria [12].

Alcune applicazioni, tra le più significative tra quelle possibili, sono:

- Misure di dimensioni e di superfici [4], [13].
- Analisi di posizione [5], [14].
- Analisi della luminosità e della sua variazione [7].
- Analisi di curve e forme e loro riconoscimento [6].
- Individuazione di punti o zone su cui effettuare delle lavorazioni [8], [9].
- Rilevazione di possibili difetti di produzione [13].
- Smistamento di oggetti diversi su una linea di produzione [11].
- Inseguimento di oggetti in movimento per consentire operazioni su di essi senza arrestarli [10].
- Comparazione di immagini [12].
- Identificazione di caratteristiche specifiche [9].
- Trasmissione di immagini [10].

Lo scopo del presente lavoro è quello di illustrare l'attività di ricerca tendente allo sviluppo delle primitive funzionali e del sistema operativo per una macchina di visione da realizzare con l'architettura del nodo MODIAC.

Tale nodo, che consiste in un sistema a moltimicrocalcolatori basato sul microelaboratore a 16 bit ZILOG Z8001, permette di realizzare un'esecuzione degli algoritmi di elaborazione delle immagini, propria delle macchine di visione, in modo altamente parallelo.

Il sistema operativo permette di gestire in modo efficiente

l'insieme di questi algoritmi che costituiscono le primitive funzionali della macchina di visione.

Il sistema operativo e gli algoritmi relativi alle primitive funzionali della macchina di visione sono attualmente funzionanti sul MINC-11 del Laboratorio di Robotica del Progetto di Intelligenza Artificiale del Politecnico di Milano.

Il sistema sarà inserito nel prossimo futuro in un nodo MODIAC realizzando così una macchina di visione ad architettura parallela.

2. PRIMITIVE FUNZIONALI DELLA MACCHINA DI VISIONE

Nell'ambito del Progetto di Intelligenza Artificiale del Politecnico di Milano già da diversi anni sono stati ideati e sviluppati diversi lavori riguardanti la robotica ed il problema della visione.

In particolare, per quanto riguarda il nostro problema, vi sono dei programmi il cui scopo è quello di realizzare una analisi dettagliata e permettere il riconoscimento di figure piane bidimensionali.

Le attrezzature di cui ci si avvale sono costiduite da :

- Telecamera allo stato solido General Electric TN2200TV. La telecamera fornisce un' immagine sotto forma di matrice di dimensioni 128x128. Gli elementi di questa matrice rappresentano il tono grigio (su 8 bit) del punto in considerazione.

- Elaboratore MINC-11 della Digital Equipment Corporation con sistema operativo RT-11 dotato di: 1 Mbyte di memoria di massa, 56 Kbyte di memoria centrale, su interfaccia DRV-11-B per il collegamento alla telecamera.
- Terminale grafico a 8 colori ISC 8001.
- Elaboratore PDP-11/34 per permettere uno sviluppo fuori linea dei programmi.

I lavori già terminati riguardano i seguenti aspetti:

A. Riconoscimento ed orientazione di figure piane bidimensionali riferite ad una coppia di assi baricentrali con una figura in posizione canonica.
B. Analisi del contorno di figure piane anche complesse.
C. Orientamento di figure piane e calcolo di un insieme di parametri significativi.
D. Analisi di connettività per la discriminazione di più oggetti presenti contemporaneamente in un' immagine.
E. Algoritmo di Hueckel per la determinazione dei contorni.

Esaminiamo ora i diversi programmi che sono stati sviluppati:

Programma A.

Questo programma è in grado di riconoscere l'immagine presa dalla telecamera. Riceve in ingresso l'immagine di un pezzo meccanico direttamente dalla telecamera ed esegue una linearizzazione in modo da poter discriminare la figura dal fondo. Operando quindi sulla figura binarizzata determina due parametri caratteristici (vedi Fig. 1, Fig. 2):

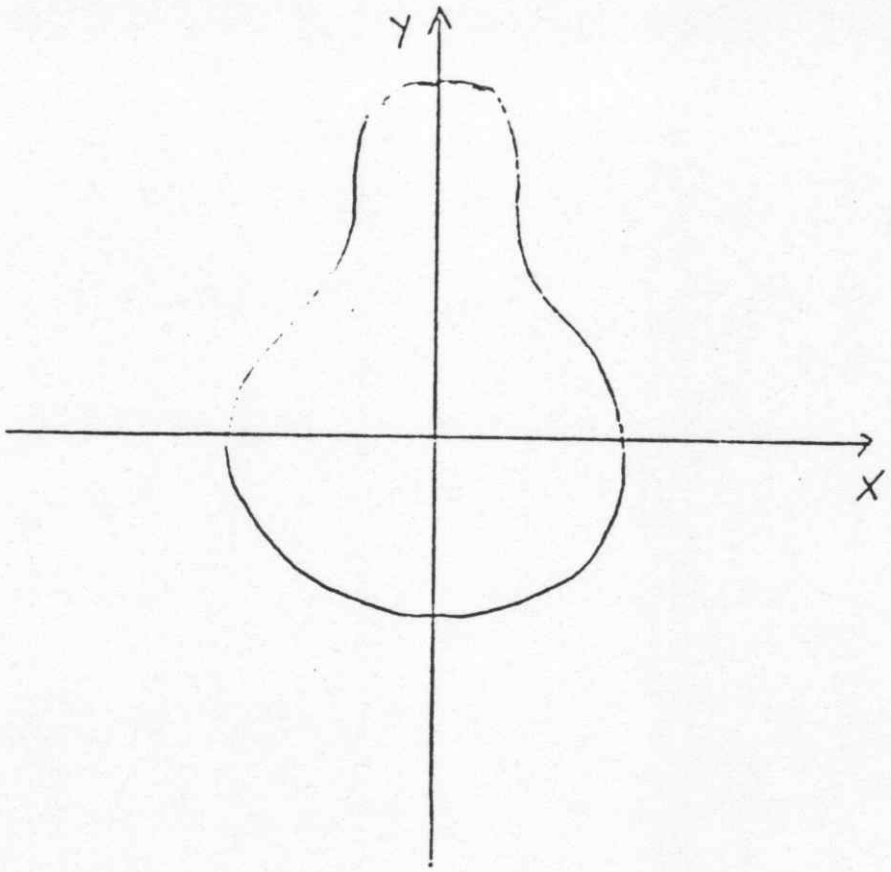


Fig. 1 - Figura in posizione canonica.

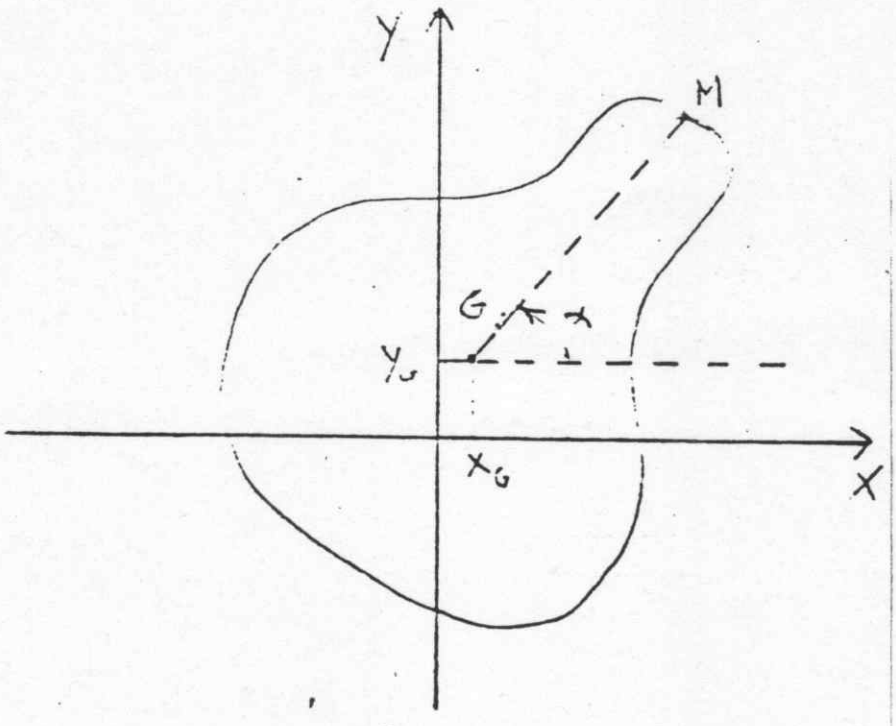


Fig. 2 - Figura in posizione generica.

- le coordinate del baricentro x_a , y_a ;
- l'angolo di rotazione del pezzo misurato tra una posizione canonica ed una generica: è l'angolo formato da un asse passante per il baricentro e parallelo con l'asse x, con la retta che congiunge il baricentro con il punto M di distanza massima dal baricentro.

In questo modo l'algoritmo è in grado di paragonare i due pezzi e può quindi riconoscere il pezzo situato in posizione generica.

Programma B.

Il programma elabora un' immagine presa dalla telecamera; al termine esso è in grado di fornire i risultati ottenuti sotto varie forme (vedi Fig. 3, Fig. 4, Fig. 5):

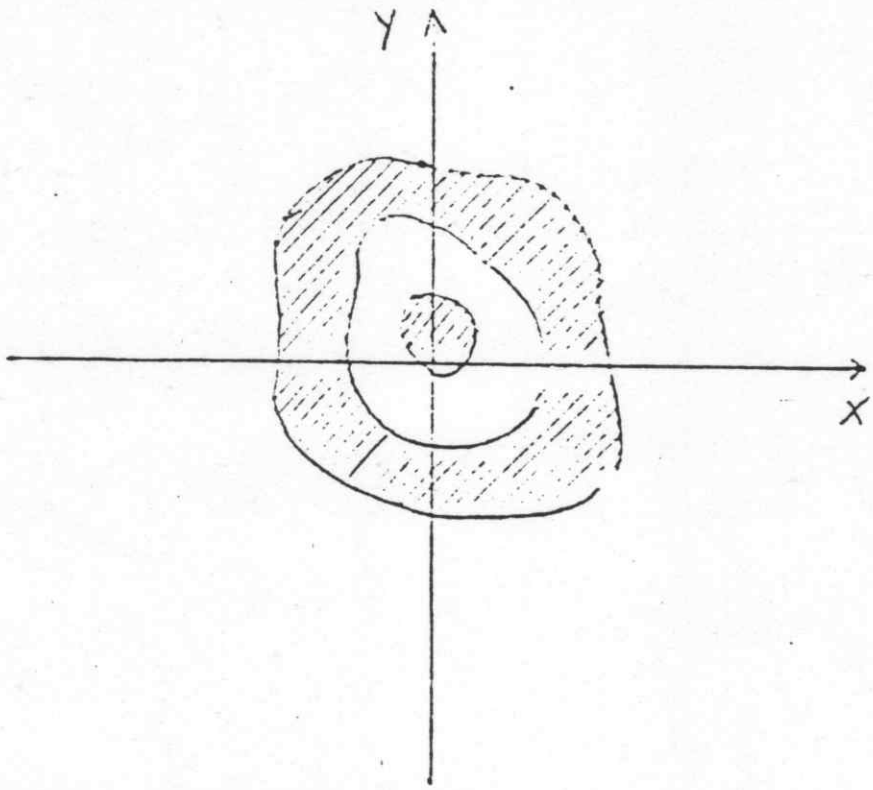


Fig. 3 - Figura originale (con livelli di grigio).

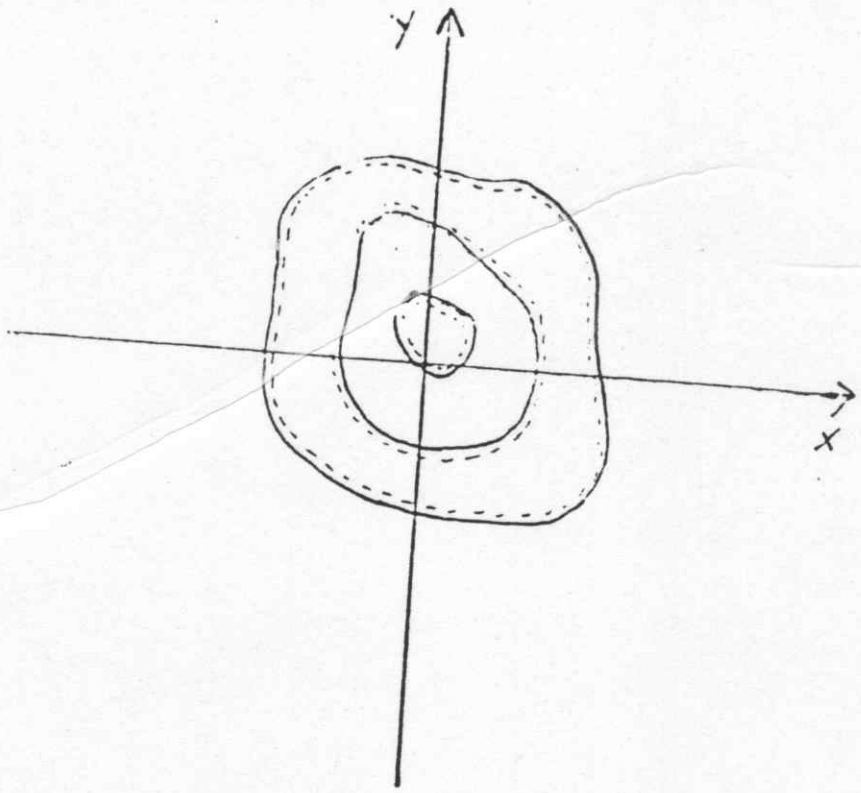


Fig. 4 - Matrice dei contorni.

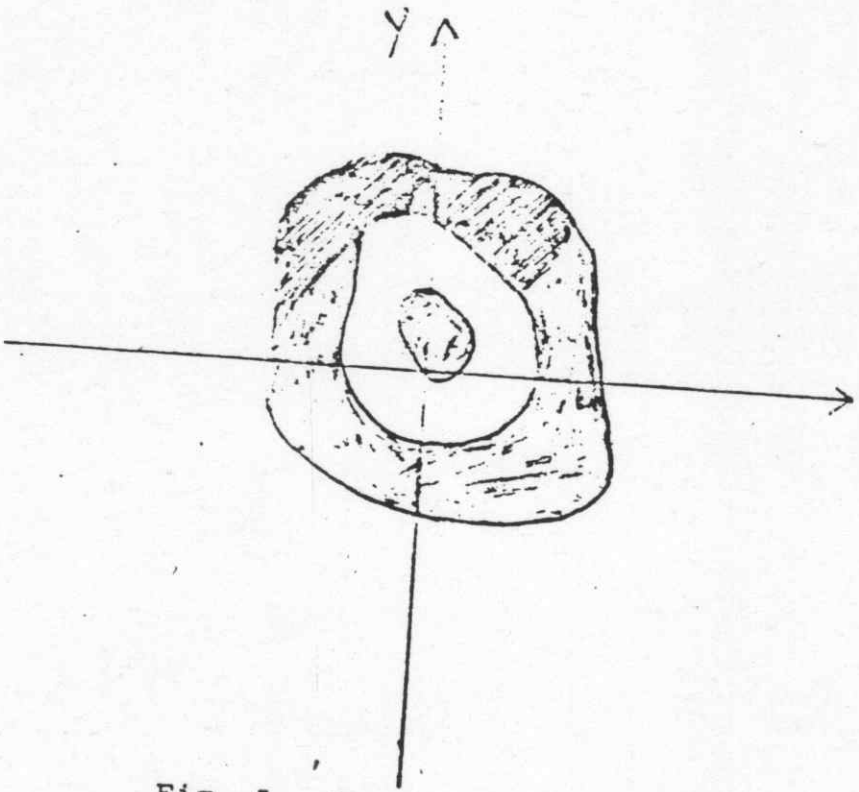


Fig. 5 - Matrice piana binarizzata.

- matrice binarizzata dell'immagine
- matrice dei contorni dell'oggetto presente nell'immagine stessa; nel caso di matrice dei contorni sono inoltre rappresentati, in maniera diversa, i punti immediatamente interni del pezzo (linea tratteggiata sulla figura) in modo da facilitare elaborazioni successive a questo programma e che ne utilizzino la matrice scelta.

Programma C.

Questo programma ha due scopi (vedi Fig. 6) :

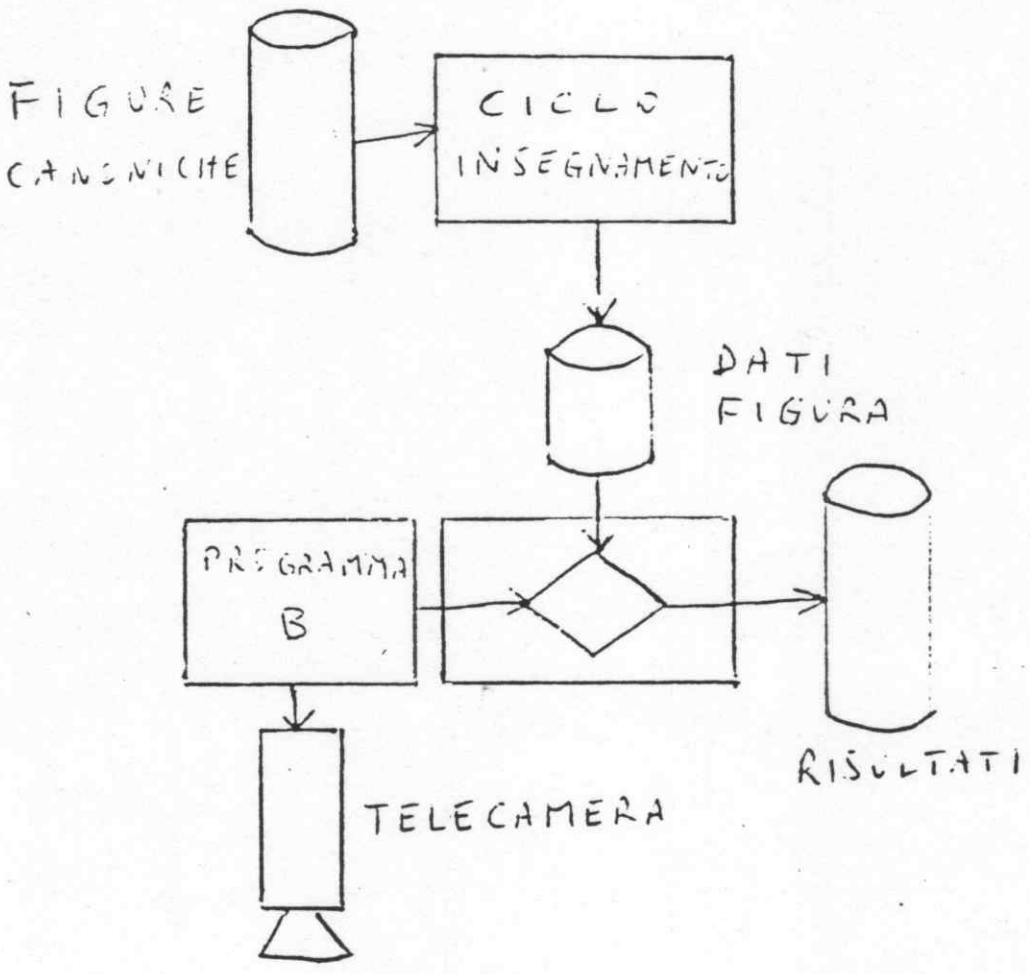


Fig. 6 - Schema del Programma C.

- determinazione dell'orientamento e della posizione del pezzo in osservazione rispetto ad una posizione canonica memorizzata precedentemente sotto forma di diagramma polare
- riconoscimento del pezzo in osservazione entro un certo numero di pezzi memorizzati precedentemente.

Potrebbe quindi sembrare che vi sia quindi una duplicazione dell'attività scelta dal Programma A; però non è questo il caso.

Infatti bisogna tener conto che nel Programma A si è badato soprattutto alla tempestività dei risultati, a scapito di una piena raffinazione dei risultati ottenuti.

Nel Programma C, invece, si è preferito dare più spazio all'accuratezza della discriminazione delle immagini, anche se a prezzo della velocità di esecuzione.

Programma D.

Si osservi la Fig. 7:

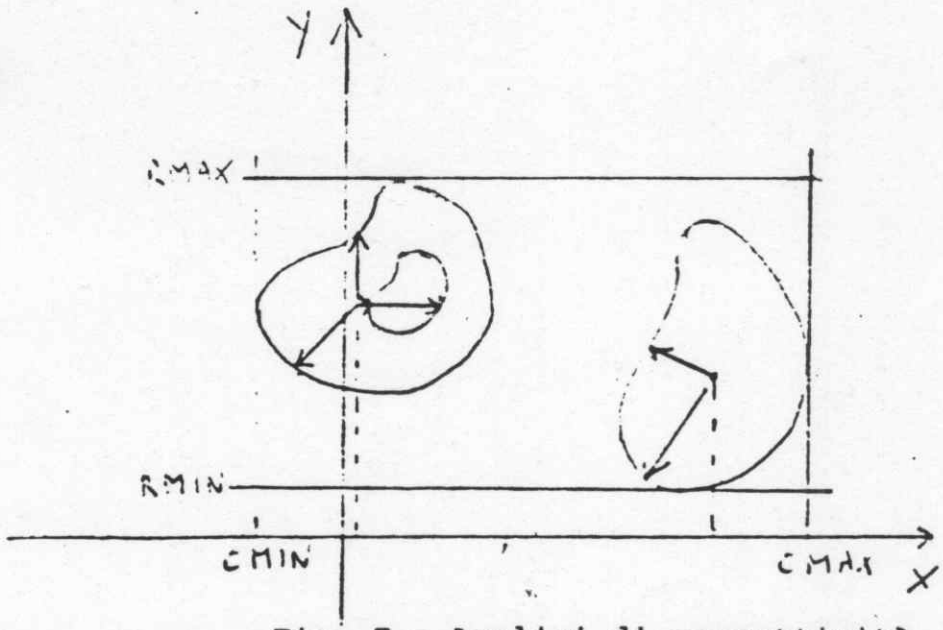


Fig. 7 - Analisi di connettività.

Questo programma esegue, sull'immagine proveniente dalla telecamera, un'analisi di connettività e l'estrazione di un certo numero di caratteristiche dei pezzi che si trovano sotto l'obiettivo.

Questo programma è quindi molto generale ed è in grado di analizzare situazioni molto complesse.

Alla fine della elaborazione si ottiene un archivio che contiene le caratteristiche geometriche dei pezzi e che può eventualmente essere usato da altri programmi per il riconoscimento di ogni singolo pezzo.

Programma E.

Questo programma, denominato Algoritmo di Hueckel, più che rivolgersi al riconoscimento dell'oggetto, si occupa dell'estrazione delle linee in cui si ha il massimo gradiente di luminosità.

Chiaramente ha degli scopi diversi dai quattro precedenti, che infatti si preoccupano di determinare l'oggetto.

Infatti adottando i criteri di questo programma è possibile ottenere delle linee anche non chiuse su se stesse (se l'immagine non ha un buon contrasto).

I vantaggi sono notevoli su immagini molto disturbate e i risultati ottenuti in questo modo sono buoni.

Uno dei limiti, invece, di questo programma è la scarsa velocità di esecuzione.

Infatti, per stabilire i salti di luminosità, vengono utilizzati degli operatori matematici che impegnano notevolmente

l'elaboratore e di conseguenza non può essere usato se si vuole una certa tempestività di risposta.

3. IL SISTEMA OPERATIVO DELLA MACCHINA DI VISIONE

Tutti i programmi visti in precedenza, allo stato attuale delle cose, non sono altro che un insieme disaggregato di funzioni interattive che comportano un certo numero di operazioni manuali per la loro attivazione.

E' chiaro che, per poter coordinare efficacemente le loro attività, risulta necessaria la presenza di un gestore che si occupi di collegare tra loro le varie parti e possa accedere all'archivio di forme.

Si tratta cioè di realizzare un sistema operativo di una macchina di visione che fornisca un certo numero di primitive di base.

Il nostro sistema operativo risulta costituito da un insieme di segmenti di programma residenti su dischetto e da un nucleo residente in un'area protetta di memoria centrale.

Ad esso si potrà fare riferimento tramite dei punti di entrata corrispondenti ai comandi.

Tali comandi sono:

- acquisizione di una immagine con 256 toni di grigio;
- acquisizione dell'immagine e memorizzazione su dischetto;
- calcolo del contorno di un oggetto;
- calcolo della matrice binarizzata;
- calcolo della soglia;

- determinazione dell'orientamento;
- calcolo dei parametri standard.

Quello che forniremo ai futuri ricercatori che ci seguiranno sarà una macchina virtuale che permetterà una gestione "integrata" dei programmi esistenti e dei dati raccolti. Il nostro sistema operativo risulta essere un'estensione, cioè l'aggiunta di un livello ulteriore, del sistema di diagrammi che costituiscono le primitive funzionali della macchina di visione.

Il tutto potrà essere considerato come il sistema di programmazione della macchina di visione che permetterà così la realizzazione (come possibile futuro sviluppo) di un interprete che veda i comandi per la macchina di visione come istruzioni di un programma.

Inoltre, tenendo conto della limitata potenza di calcolo del calcolatore MINC-11 e tenendo presente l'utilità di poter usufruire di un calcolatore più potente (nel caso particolare si tratta del PDP-11/34) per lo sviluppo dei programmi, si è deciso di rendere disponibili, tramite il sistema operativo, una serie di programmi di utilità che permettono il trasferimento, in modo pressochè invisibile, di dati e programmi dal PDP-11/34 al MINC-11 e viceversa.

4. DESCRIZIONE DELL'ARCHITETTURA DEL SISTEMA DI PROGRAMMAZIONE

L'architettura del sistema è stata individuata tenendo conto sia delle problematiche esistenti sia dei programmi già sviluppati sia di quelli di previsto sviluppo futuro.

I processi elaborativi che si collegano al nostro sistema operativo sono stati divisi in due gruppi:

a) segmentazione;

b) riconoscimento.

Col termine segmentazione si intende l'identificazione di caratteristiche essenziali per l'oggetto (come ad esempio il contorno); con il termine di riconoscimento si vuole identificare l'attività di confronto fra il pezzo in osservazione ed un pezzo canonico di riferimento.

Si è stabilita una sequenza orizzontale di processi, delimitati da due interfacce fissate A e B come illustrato in Fig. 3 che segue.

Nella figura si sono inoltre posti in evidenza i programmi di servizio (od utilities, vedi Appendice) che verranno usati per i trasferimenti di dati e di programmi tra i calcolatori MINC-11 e PDP-11/34.

Anche questi programmi rispettano le interfacce fissate, da noi definite, e si presentano, ad un osservatore esterno, assolutamente indipendenti dal calcolatore usato. Infatti, questi programmi di servizio e queste interfacce

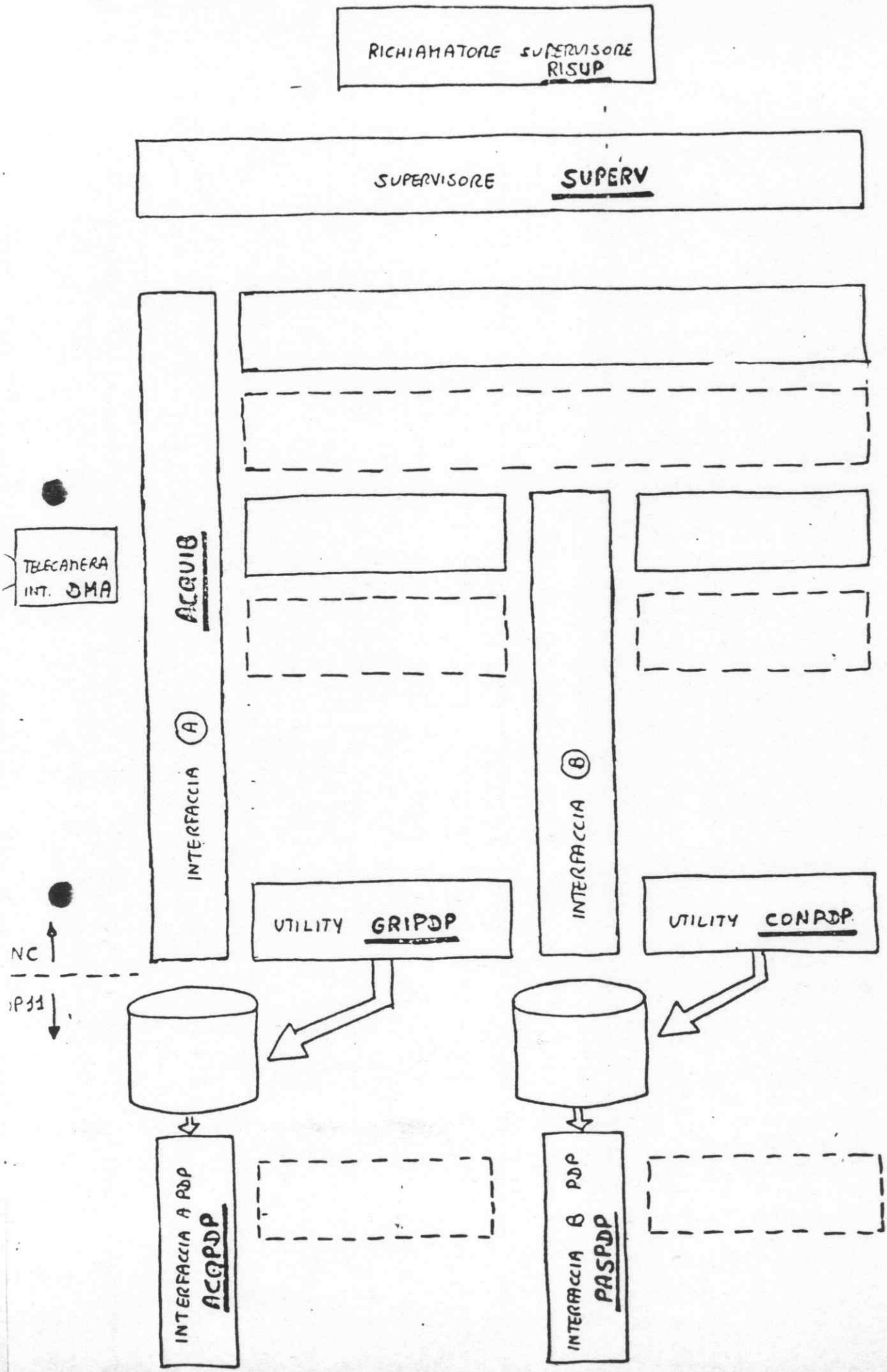


Fig. 8 - Programmi di servizio.

fissate servono per togliere all'utente tutti quei problemi legati alla diversità della macchina, realizzando così una struttura modulare in cui l'aggiunta di un nuovo modulo non crea problemi di modifiche per i programmi preesistenti.

5. REALIZZAZIONE DEL COLLEGAMENTO TRA TELECAMERA E PROGRAMMI

Esaminiamo ora il procedimento di acquisizione dell'immagine con 256 livelli di grigio. Questa funzione era stata realizzata sino ad ora mediante l'esecuzione di un programma.

Questo tipo di realizzazione, pur a fronte del suo basso costo, comportava tuttavia una notevole lentezza di trasferimento dei dati fra telecamera e calcolatore e causava, inoltre, notevoli problemi di definizione dell'immagine.

L'immagine risultava molto disturbata a causa del tempo di esposizione dell'oggetto che risultava eccessivamente lungo (per acquisire un' immagine era necessario circa un secondo e la telecamera si saturava facilmente).

Recentemente abbiamo sviluppato e realizzato una interfaccia software/hardware che ha permesso di collegare la telecamera all'elaboratore tramite una scheda di DMA (vedi Fig. 9).

La parte hardware sincronizza la telecamera ed il circuito di DMA e la parte software (è costituita da un programma, scritto in linguaggio assemblatore) gestisce il collo-

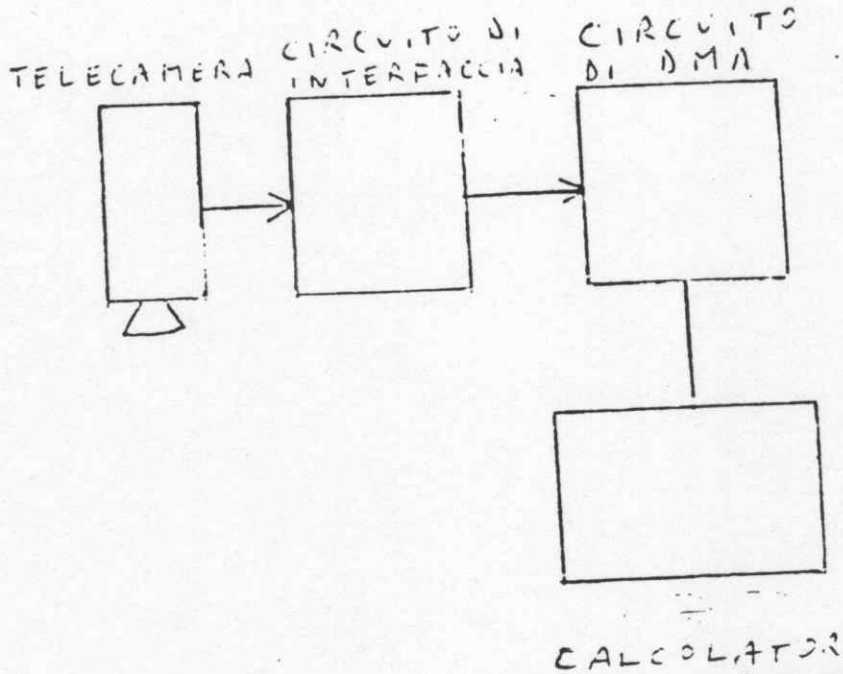


Fig. 9 - Collegamento diretto tra telecamera e calcolatore.

quio tra la telecamera ed il programma utente.

L'acquisizione tramite DMA è molto veloce (1/25 sec.) e questo ha consentito un notevole miglioramento dell'immagine.

Ciò può permettere anche di memorizzare immagini di pezzi in movimento illuminati con dei flash.

L'utilità di questo fatto è evidente se si considera che i pezzi, di solito, viaggiano su dei nastri trasportatori, ad una certa velocità; non è quindi pensabile di fermare il nastro ogni volta che uno di essi passi sotto l'obiettivo (vedi Fig. 10).

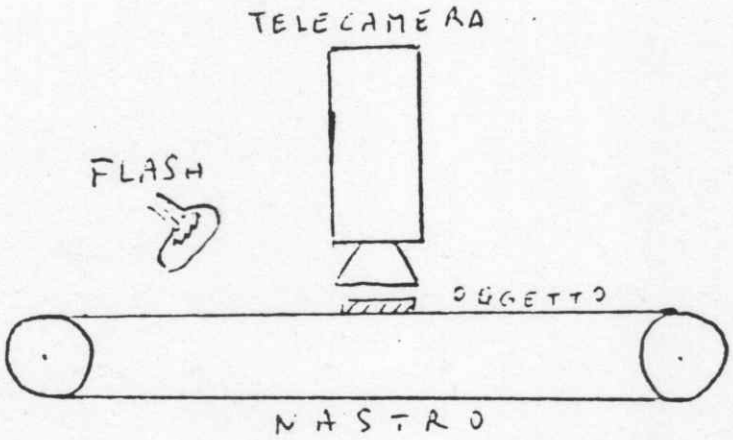


Fig. 10 - Nastro trasportatore e telecamera.

6. DEFINIZIONE DELLE INTERFACCE

Il primo passo effettuato per realizzare l'integrazione dei vari programmi disponibili è stato quello della definizione di interfacce fissate per rendere tali programmi intercomunicanti.

Sono state individuate le seguenti due interfacce :

a) matrice a livelli di grigio (per i programmi che realizzano la segmentazione);

b) matrice binarizzata (per i programmi di riconoscimento ed orientazione).

Matrice a livelli di grigio.

Questa interfaccia è stata praticamente imposta dall'hardware

della telecamera. Infatti essa fornisce un quadro di 128x128 punti (pixels).

L'interfaccia è quindi una matrice di numeri di 8 bit, ciascuno, che assumono un valore compreso tra 0 e 255 relativo al livello di grigio del pixel corrispondente.

Quello che il programma utente deve fornire al programma di acquisizione è l'indirizzo di memoria dove mettere i valori della matrice.

Per minimizzare l'occupazione di memoria, è opportuno che il programma che richiede l'acquisizione tratti la matrice come se fosse composta da dei caratteri, e non da dei numeri interi. Questo perchè la codifica di un carattere occupa un byte (ottetto di bit) mentre quella di un intero occupa una parola (cioè 16 bit) e quindi, globalmente,16 Kbyte invece di 32 Kbyte.

Matrice binarizzata.

Questa interfaccia consta di una matrice di 128x128 caratteri. Tali caratteri rappresentano lo sfondo ed il pezzo come definiti dal primo programma di segmentazione realizzato.

Questa interfaccia è tuttavia realizzata mediante un'archivio di transizione sul quale accedono i programmi ai due livelli. Si è notato però che tale metodo di passaggio è piuttosto inefficiente in termini di velocità e pertanto verrà realizzato un metodo che permetta l'impiego esclusivo

della memoria centrale. Questo non sarà però possibile, se non verrà disabilitato il sistema operativo del MINC-11, perchè esso non ha la possibilità di salvare i dati nella memoria centrale.

7. CONCLUSIONI

Nel presente lavoro abbiamo illustrato le primitive funzionali ed il sistema operativo della macchina di visione che è stata sviluppata presso il Laboratorio di Robotica del Progetto di Intelligenza Artificiale del Politecnico di Milano.

Nel prossimo futuro il sistema di programmazione completo sarà portato dall'attuale calcolatore, il MINC-11 della DEC, al nodo MODIAC, costituito da un sistema a multimicrocalcolatori Z8001 della ZILOG.

B I B L I O G R A F I A

[1] Somalvico, M. Robotica. CLUP - Cooperativa Libraria Universitaria del Politecnico. Milano, 1977.

[2] Somalvico, M. Robotica. Temi di Ricerca 1982. CLUP - Cooperativa Libraria Universitaria del Politecnico. Milano. (In pubblicazione).

[3] Somalvico, M., Viviani, C." La visione nella "Computer Aided Manufacturing" (CAM)". Automazione e Strumentazione, n. 4.Aprile 1981.

[4] Kammenos, A., Panaijotiis, D.,"Performance of polar coding for visual localization of Planar objects". Laboratoire de treatment des signaux du Politechnique Federale de Lausanne.

[5] Martini, P., Nehr, G." Recognition of angular orientation of objects with the help of optical sensor". Inst. Für Informatik, University of Karlsruhe, Karlsruhe, West Germany.

[6] Shirai, I."Edge finding, segmentation of edges and recognitions of complex objects". Advance papers of the Fourth International Joint Conference on Artificial Intelligence. Tbilisi, Georgia, USSR. 1975.

[7] Rosenfend, A., Weszka, J.S."Picture Recognition". In Digital Pattern Recognition, pp. 144 - 147. Academic Press. New York. 1972.

[8] Duda, R., Hart, P.E"Contour following." In Pattern Classification and Scene Analysis, pp. 290 - 293. Wiley Inc. New York. 1973.

[9] Rosenfend, A."Picture processing by computer". Computer Survey. September 1969.

[10] Rosen, C.A."Machine vision and robotics industrial requirements". SRI Technical Note 174.

[11] Grasselli, A. Automatic interpretation and classification of images. Academic Press. New York. 1976.

[12] Koskigen, K., Niemi, A."Object recognition and handling in an industrial robot system with vision". Proceedings Eight ISIR. Stuttgart, West Germany. 1978.

[13] Hueckel, M.M."An operator which locates edges in digitized pictures". Journal of ACM, Vol. 18, pp. 113-125. 1971.

[14] Hueckel, M.M. "A local visual operator which recognizes edges and lines". Journal of ACM, Vol. 20, pp. 639-697. 1973.

APPENDICE. DESCRIZIONE DEI PROGRAMMI DI SERVIZIO

A. UTILITY GRIPDP

L'utility GRIPDP è un programma PASCAL che acquisisce una figura (matrice di grigi) tramite la subroutine ACQUIB e la memorizza su dischetto per poi essere utilizzata sul PDP-11/34 per lo sviluppo dei programmi al livello 1 (segmentazione).

L'archivio utilizzato è un archivio di caratteri che rappresenta l'immagine esatta della memoria centrale (a meno di alcune codifiche resesi necessarie per evitare i problemi inerenti ai caratteri di controllo).

B. UTILITY ACQPDP

L'utility ACQPDP è un programma PASCAL che svolge sul PDP-11/34 la stessa funzione che la subroutine ACQUIB svolge sul MINC-11; cioè si acquisisce una matrice di 128x128 elementi di toni di grigio e la si restituisce al programma chiamante.

Essa viene utilizzata durante lo sviluppo dei programmi a livello 1 (segmentazione) sul PDP-11/34.

C. UTILITY CONPDP

L'utility CONPDP è un programma PASCAL che acquisisce una figura al livello 2 e la memorizza su archivio mediante dischetto, per poi essere utilizzata sul PDP-11/34 per lo sviluppo dei programmi a livello 2.

D. UTILITY PASPDP

L'utility PASPDP è un programma PASCAL che viene utilizzato durante lo sviluppo dei programmi a livello 2 (riconoscimento) sul PDP-11/34.

Restituisce al programma chiamante la matrice che si trova al livello 2 sul MINC-11.